シャープ MZ-1500

テクニカルノート

パソコンの仕組みを探る

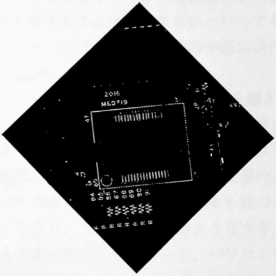
日本マイコンクラブ監修

ラジオ技術選書151

シャープ MZ-1500

テクニカルノート

パソコンの仕組みを探る



日本マイコンクラブ監修

入門者にとっても読みこなせるものに

高橋三雄

ひところ，文明の尺度として，文盲率とか電話の普及率あるいは新聞の購読部数などが使われました．しかし現在では，国民一人あたりのマイコン使用個数がもっともよく文明の水準をあらわしているのではないでしょうか．といっても，マイコンの多くは製品に組込まれており，それと意識しなくとも，誰でも使うことができます．この点からはむしろ，同じマイコンの応用製品であるパソコンの方が文明度指標としてすぐれているかもしれません．パソコンは機械があっただけでは何もしてくれません．やりたい仕事なり目的をもち，また，そのためのソフトを手にする必要があります．パソコンを使いこなすためには，文盲ではだめなことは当然ですが，さらに，新しい道具をものにしようという強い意欲を持つことが大事です．私は現在アメリカの大学で研究を行なっていますが，ときどき，こちらの会社を訪問する機会があります．その時にいつも感心するのは，40過ぎの中年のおばさんが一生懸命パソコンを操作し，結構，むずかしそうなソフトを使いこなしていることです．「どこで勉強したのか？」と聞いてみると，ソフトの説明書をみて自分で勉強したと答えます．こんなところにアメリカにおけるパソコン活用のすその広さを感じさせられます．もちろん若者の世界である大学ではパソコンは空気や水のような存在で，いまではパソコンの知識がないと講義を受けることさえできないといっても言い過ぎではありません．もちろんこれはビジネススクールの話で，理科系の学部にいけば，パソコン必要度はさらにエスカレートすることになるでしょう．

さて，日本もいつのまにかパソコンの普及台数は200万台を越えたといわれます．これはアメリカに次いで，世界第二位であることはいうまでもありません．パソコンも一時は単なるブームでやがてすたれるともいわれました．しかし，いまはそんなことをいう人はほとんどいないでしょう．最先端のパソコンであれば５，６年前の大型コンピュータのソフトをそのまま動かすことさえできます．人によっては10年も経つと，一つの部屋を占有するようなコンピュータは珍しくなり，すべて机に乗るような大きさになると予想しています．いずれにしても，ほとんどあらゆる人々にとって，パソコンはそれぞれの必要度に応じて身につけるべき道具になってきているといえます．このことはアメリカで生活しているととくに強く感じられます．

現在アメリカではつぎつぎと新しいパソコンやソフトが発売されています．そして，それらがうまくユーザーに受け入れられるかどうか，そのもっとも重要な要因の一つとして使いやすさが強調されています．たとえば，ハードウェアでいえば，キーボードの配置やタッチでさえ，評価の大きなポイントですし，メニューやマウスなど，パソコンとのコミュニケーションをよりよくする手立ては高く評価されています．また，統合ソフトといって，いろいろな仕事が，いつでも簡単に選択し，実行できるように工夫されたソフトが広

く利用されています．このように，ユーザーの立場を最優先するハードとソフトの開発の必要性がいずれは日本でも認識されてくることになるでしょう．

パソコンは人によってその使い方に大きな差があります．しかし，現在ではコンピュータとしてのパソコンに対する興味よりも，パソコンで何ができるのか，あくまでそれを道具として使おうとする人々の数が大半を占めるようになってきているといってよいでしょう．こうした人々を対象とした，使いやすいハードあるいはソフトさらには各種の参考書が求められます．本書は，シャープのMZ-1500を使い，はじめてのユーザーにとっても十分に読みこなせることを第1にしています．ここで学ぶことがらは，ほかの機種でも大いに参考になるはずです．本書をきっかけに，パソコンというすばらしい道具をいつも身近に置かれるようになることを期待したいと思います．

なお，本書の執筆にあたっては，当クラブのメンバーである野々山隆幸氏（横浜市立大学助教授），加藤隆明氏（芝浦工業大学）はじめ，シャープの若い開発者（高井靖之，飯塚豊，岡本慎二，喜多敏郎，湯下良一，辰巳卓，竹本正人，高橋一成，十河祥介，横井寿治，杉本達也）の諸氏のご協力をいただきました．

最後に出版に際して，㈱ラジオ技術社 境 輝正氏，鈴木哲哉氏に大変お世話になり感謝致します．

1984年12月ロサンゼルス，USCにて．（成蹊大学教授　日本マイコンクラブ理事）

MZ-1500テクニカル・ノート パソコンの仕組みを探る
＊目次＊

第1章 MZ-1500の開発とその背景

1-1：コンピュータの誕生とその発達 …… 8
⑧電子計算機の登場 ⑪コンピュータの基本構造 ⑬マイコン，そしてパソコンの登場

1-2：MZシリーズとMZ-1500 …… 16
⑯MZの誕生 ⑲MZ-1500の特徴

第2章 パソコンの基礎

2-1：パソコンの構成 …… 28
㉙中央処理装置 ㉚入力装置 ㉛出力装置 ㉜入出力装置

2-2：ソフトウェアの役割 …… 34
㉞ソフトウェアの歴史 ㉟ソフトウェアの概念 ㊲IPL ㊳モニタ
㊴BASICインタープリタ

2-3：LSI素子について …… 46
㊼Z80CPU ㊾Z80PIO ㊿Z80PIOのアーキテクチャ 52 Z80PIOのプログラミング
56 Z80SIO 61 8255 62 8253 65 CRT，メモリコントローラ 65 PSG

第3章 ハードウェア

3-1：MZ-1500のソフトウェア …… 70
70 BASICについて 74 IPLについて 139 モニタについて

3-2：MZ-1500のハードウェア …… 152
152 MZ-1500の構成 154 RAMファイル 160 CRTC 165 QDとQDコントローラ
174 QDハンドラー 180 カセットインターフェイス 181 キーボードスキャン
182 プリンタインターフェイス 188 PSG

第4章 MZ-1500の応用

4-1：PCGの使いこなし …… 196
196 PCGの重ねあわせ 199 使用上の注意

4-2：パターンエースの使いかた …… 201
201 エディタの起動 203 大きな絵を描く

4-3：ミュージックエースの活用 …… 206
206 音符データの入力方法 208 サウンド作りに挑戦 210 実際の作成例

APPENDIX

Z80命令表 …… 212　MZ-1500の全回路図 …… 216

第1章 MZ-1500の開発とその背景

1-1：コンピュータの誕生とその発達

電子計算機の登場
コンピュータの基本構造
マイコン，そしてパソコンの誕生

1-2：MZシリーズとMZ-1500

MZの誕生
MZ-1500の特徴

1-1 コンピュータの誕生とその発達

1-1-1 電子計算機の登場

1946（昭和21）年，世界最初の電子式コンピュータENIACが出現したことは，計算機史上きわめて大きな意義を持っています．それまでの自動計算機が，継電器（リレー）を用いて構成され，演算に数秒～10数秒の時間を要したのに対し，真空管の持つスイッチング作用を利用したENIACは，機械的に動く部分がないため高速の処理が可能となり，10進数10けたの加算を0.2 mS，乗算を3 mSで行いました．この値は今日のマイコンがソフトウェアで実行するのと比べても，それほど遜色はないでしょう．しかし，記憶装置の容量は現今のコンピュータと比べるとぐっと少なく，10けたの10進数を18個記憶できるだけでした．使用された真空管の本数は18,800本，120 kWというとてつもない電力を消費しました．写真1·1にENIACの外観を示します．

ところで，このマシンで，まっさきに行った計算はなんでしょうか．これは現在のわれわれにも大変興味のあるところですが，じつは弾道計算が行われたのです．弾道計算とは，砲弾の軌道計算からその着地点を予測することで，現代ならばさしずめミサイルの軌道計算といったところでしょう．当時，砲弾が1分間に飛ぶ道筋を計算するのに，手計算で10数時間を要したのですが，ENIACはこれをたった30秒でやってのけ，タマの飛ぶよりも早く計算できるということが，大変な話題になったそうです．ENIACは，戦時研究にこそ間に合わなかったのですが，もともと第二次大戦中，米国海軍のために，こうした計算が目的で開発されたものでした．

制御上の問題

ENIACの欠点は，大量に使用された真空管の保守にあたっての煩雑さと，記憶容量の少なさにありました．しかし，それよりももっと大きな問題が横たわっていました．それはコンピュータの制御にかかわる重大な問題でした．

現在，われわれの間では，コンピュータに何らかの処理を行わせる場合，命令を書き連ねたプログラムが必要で，実行にあたってそれを内部の記憶装置の中に格納しておくということは常識になっています．いわゆる**プログラム内蔵方式**です．しかし，こうした考え方は最初からあったわけではありません．

ENIAC以前の継電器式マシンでは，幅の広い紙テープ（現在の8単位テープより広かった）に一列に穿孔された命令を，1個読んでは実行し，また次の命令を読むということの繰返しでした．これが，低速度のマシンの場合はけっこううまくいっていました．ところが，電子式のENIACがこれでは，あまりにも時間がむだになります．なぜなら，演算装置は毎秒5,000回も加算できるのに，動作の遅い機械式の読取り装置から，命令をそのつど読んでいたのでは，演算装置は大部分の時間，遊んでしまうことになるのです!!

そこで，ENIACではプラグボードというものが考えられました．これは，配線盤上でプラグを差しこむことにより，演算に応じた配線を行おうと

写真1·1　ENIAC（スペリー・ユニバック提供）

するものでした．これならば速度に関するかぎり何ら問題はありません．これはその後のアナログコンピュータにも継承された方式です．

しかし，このプラグボード方式にも大きな欠点がありました．それは，問題を解き終って次の問題に移るとき，配線をすっかりやり直さなければならないことです．配線に手間どり，その間マシンが使えないのでは，いくら計算が早くても（いや，早ければ早いほど）やはりマシンが遊んでしまうことになります．結局，この方式も本質的な問題解決にはなり得なかったのです．

ノイマン型コンピュータへ

こうした制御上の難点をみごとに解決し，今日のコンピュータ時代の基礎を築いたのはフォン・ノイマン（Von Neumann）です．彼は次のようなプログラム内蔵方式というものを提案しました．

一般にコンピュータの処理内容は，データの入出力，データの移動，各種演算と桁ずらし（シフト），それに判断といった，比較的単純な操作に分解でき，その種類はそれほど多くありません．そこで，こうした操作の一つ一つを，たとえば加算ならば01，減算ならば02という具合に，数値で表わす（符号化する）ことにします．一方記憶装置は一連番号を付け，特定のデータを指す場合，何番地に入っているデータ，と指定するものとします．そうすると，たとえばある数にある数を加える場合，その番地を指定すればよいわけです．

このように，すべての操作を数値化すると，処理内容は一連の数値の集まりとなり，これらをデータとともに記憶装置の中に入れてしまいます．こうすれば，コンピュータは記憶から操作を表すデータを一つ一つ順番に取出してきて，それが何の操作であるかを解読し，必要な演算を行なえばよい，というのです．

したがって，この方式では処理の過程はコンピュータ内部で自動的に制御されることになります．この場合，制御装置は電子回路ですから，速度の点では何も問題はありません．それにプログラムが変っても，結線をし直すわけではないからマシンの構造は変りません．また，プログラミング作業はマシンから離れて独立してできます．このようにして，プログラム内蔵方式は速度と制御上の二つの難点をみごとに解決したのです．そこで，その後のコンピュータでは，この**プログラム内蔵，逐次処理方式**が基本原理として確立し，現在に至っているわけです．現時点でのコンピュータを，ノイマン型コンピュータと呼ぶのはこのためです．

ところで，フォン・ノイマンに関しては一つのエピソードがあって，彼が第二次大戦中，戦時研究に動員され，弾道学の研究をやっているらしいという情報がわが国にも伝えられていました．そこで日本の関係者の間では，ノイマンはこれまでにも数学や物理学の分野に新しい手法を導入してきた人だから，弾道学という古い学問にもきっと思いがけない新機軸を打ち出すに違いないと考えられていたそうです．

だが，ノイマンの関心事は，古くからある弾道学ではなく，別のところにありました．それでも，ENIACが最初にやったのは弾道の計算だったので，彼が弾道学を研究しているということは，あながち間違いではなかったわけです．たしかにノイマンは新機軸を打出したのです．

ENIAC以後のコンピュータ

ENIAC以後，1949年にEDSAC，1950年にEDVACが作られました．これらは，いずれもプログラム内蔵方式のコンピュータです．また，1951年には，UNIVAC Iが作られました．これが商用コンピュータの第1号です．

これらのコンピュータは，いずれも真空管式でした．そこで，主要素子として真空管が使用されたこの時代，すなわちENIACが出現した1946年から1955年までを第1世代と区分します．

わが国では，FUJIC，TACなどがこの世代に属しますが，日本では真空管式コンピュータはあまり多くありません．当時，パラメトロンという素子が発明されたので，これを使ったコンピュータがつぎつぎに作られたのです．そして，すぐに第2世代に移行していきました．

この頃，1948年にショックリイ（W.B.Shockley）らによって発明された，**接合形トランジスタ**は，幾多の改良が加えられ，1955年頃からコンピュータに使用されるまでになってきていました．1956～1963年の第2世代はトランジスタの時代です．

この時期，IBM 140シリーズ，UNIVAC-490，BURROUGHS B5000，わが国初の商用コンピュータNEAC 2201など，特色あるマシンがつぎつぎに作られました．

また，この時代にはプログラミング言語がいくつか開発されています．代表的なものはFORTR-

年代	おもな内容
BC500頃	ピタゴラス，板の上に砂をまいて計算
1617	ナピア（英），「ナピアの骨木」発明
1632	オートレッド（英），計算尺を発明
1642	パスカル（仏），加算機を発明
1694	ライプニッツ（独）パスカルの計算機を改良した四則演算機を発明
1812	バベッジ（英），差分機関を着想
1890	ホレリス，パンチカードシステム完成
1923	日本でパンチカードシステム使用される
1944	自動計算機MARK1完成
1946	ノイマン（英），プログラム内蔵方式の概念提唱
	最初の電子式コンピュータENIAC完成，コンピュータ第1世代始まる
1947	トランジスタの原理発見される
	日本で分類集計機完成
1949	最初のプログラム内蔵式コンピュータEDSAC完成
1950	世界初の商用機UNIVAC1，米統計局へ
1952	日本初の電子式コンピュータETLMARKI完成
1954	日本でパラメトロン発明される
1955	輸入コンピュータ第1号，UNIVAC120（真空管式）導入
1956	日本初の電子式コンピュータFUJIC完成
	日本初の計算サービス始まる
	コンピュータ第2世代にはいる
1957	日本でMUSASINO－1（パラメトロン式）完成
1959	東京大学TAC完成
	国鉄，座席予約システム開始
1964	第18回東京オリンピック大会で競技運営のオンラインシステム活躍
	国鉄「みどりの窓口」開設，日本航空，国内線の座席予約システム開始
	IBM，システム360発表
	コンピュータ第3世代始まる
1965	国産第3世代機出そろう
	オンライン・バンキングシステム開始
1966	スーパーコンピュータ開発始まる
1967	衆議院選挙にコンピュータ活躍
1968	TV番組自動編集にコンピュータを使用
1970	プッシュホンの電話計算サービス開始
	コンピュータ第3．5世代始まる
1971	最初のマイコンMCS－4誕生
1972	ダイクストラ，構造化プログラミングを提唱
1975	第12回国勢調査にマークシート登場
1976	スーパーコンピュータCRAY1
1977	APPLEII，TRS－80，PET登場，日本マイコンクラブ設立
1980	国産8ビットパソコン発表あいつぐ
1981	米国防総省Ada開発
1982	国産スーパーコンピュータ出現
1983	APPLE Lisa
	「マイコン利用者認定試験」始まる
	IBMコンパチブル機のソフトウェア問題発生

表1・1　コンピュータの進歩

参考文献
1. 小林三四郎：「コンピュータの歴史」マイコンサーキュラVol.9 No.2(日本マイコンクラブ)
2. 猪飼国夫編：「3，4級マイコン入門」(CQ出版)

AN，COBOL，ALGOLですが，前の二つは現在でも広く使われています．そしてALGOLの流れをくむのがPascalです．

第2世代に続く第3世代のさきがけになったのはIBM 360シリーズで，この頃からコンピュータに**集積回路**（IC）が使用されるようになってきました．この時代には，前記IBM 360シリーズやUNIVAC 9000など多くのマシンが作られました．そして，1965年には記憶装置の容量を除いて現在のパソコンにほぼ匹敵する，初のミニコンPDP-8も作られています．また，最初の4ビットマイクロプロセッサ4004が誕生し，**マイクロコンピュータ**MCS-4が生まれたのもこの時代のことです．

第3世代（1964～1969年）の特徴は，ICの使用によりコンピュータが小型化し，高性能化がより促進されたことです．これにより，多くの機能を持った**オペレーティングシステム**がつぎつぎに開発されました．また，大型コンピュータを多数の使用者が独立して使えるようにした，共同利用システムであるタイムシェアリングシステム（TSS）も現われています．

この第3世代の次は，超LSIの第4世代（1980年～）です．そして，それに続く**人工知能**の第5世代に向けて開発が進められています．**表1・1**はコンピュータの進歩のあとを概観する年表です．

1-1-2 コンピュータの基本構造

コンピュータは，図1・1のように中央処理装置，記憶装置，入出力装置をもって構成されています．この基本構造に関する限り，大型高速のスーパーコンピュータもパソコンも同一と考えられます．また，最近では産業用機器や家電製品にまでマイコンが組込まれていますが，こうしたものもコンピュータである以上，基本的には同じ構造を持っています．

中央処理装置は，Central Processing Unitといい，その頭文字をとって通常CPUと略称されています．これがシステムの中枢です．内部の演算装置は，一般にALU（Arithmetic Logic Unit）と呼ばれる**演算回路**とデータの一時記憶としての**レジスタ**，それに1ビットの情報を蓄える**フラグ**類から成り，算術演算や論理演算などを行っています．そして，数値の演算，比較後は，その結果に応じてフラグ類が変化します．また，制御装置はコンピュータ全体の機能を司どるもので，記憶装置内の命令を取出し（フェッチ）て解読（デコード）し，各部の動作を制御します．

マイコンに使われているマイクロプロセッサは，このCPUの機能をICとし，1ないし数個のLSI化したものです．そして，とくに1個だけでCPUの本質的動作を行うものをワンチップCPUといいます．現在，もっともよく知られているZ 80，6809，8086，68000など，すべてこのタイプです．ただし，世間ではマイコンという言葉を，しばしばマイクロプロセッサという意味でも使用します．したがって，マイコンという言葉が出てきたら，それがコンピュータとCPUのどちらを指すかに十分注意してください．

CPUが実行する命令を格納する記憶装置は，一

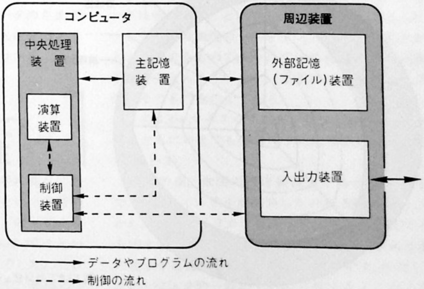


図1・1
コンピュータの基本構造

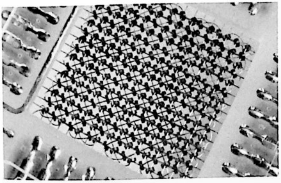


写真1・2　磁心記憶装置

般に主記憶と呼ばれ，現在は半導体LSIです．ENIACには，現在と同じ意味での主記憶というものはありませんでしたが，その後英国で開発されたEDSACでは，水銀タンク中の超音波遅延による記憶装置が考案され，わが国でも東京大学のTACは陰極線管（CRT）記憶装置というものを使用していました．

このように，コンピュータの揺籃期には先人の努力によりさまざまな記憶装置が作られましたが，記憶装置としての確固たる地位を最初に占めたのは**写真1・2**に示す磁心記憶装置です．一般に，フェライトコアの環に導線を通して電流を流すと，一方の向きに磁化されるので，どちらの向き（磁石でいえばNかSか）に磁化されたかで情報を記憶するのがこの装置の原理でした．

磁心記憶装置は，電源を切っても記憶内容が失われず(いわゆる不揮発性)，読出しと書込みの時間も（当時としては）短かったので，比較的長い期間コンピュータの主記憶として使われました．しかし，ICが出現するに及んで，その地位を半導体記憶装置に譲り渡したわけです．

どうも話がコンピュータの基本構造から脱線したようです．本題に戻って入出力装置について書きましょう．

コンピュータ本体は電子装置ですが，入出力装置には機械的に動く部分があるため多くの問題があります．その最大のものは，コンピュータ本体との動作速度の著しい違いです．したがって，入出力装置の運用の善し悪しで，システムの速さが左右されることになります．このため大型機のシステムでは，ハードウェアおよびソフトウェアにさまざまな工夫がこらされています．こうしたことは専門書で勉強してください．

一般に，コンピュータに接続されている装置としては，入出力装置のほかに**外部記憶装置**（ファイル装置），**通信制御装置**といったものがあります．しかし，これらも広い意味では入出力装置になっています．こうした装置を通常，総称して**周辺機器**と呼びます．

コンピュータの階層構造

コンピュータは，一般に**図1・2**のような階層構造を成しています．階層構造の概念は，初心者にとってちょっと理解しにくいかもしれませんが，要するにコンピュータも，平安朝の貴婦人のように，幾重にも衣裳をまとっているということです．あるいは，殻をむくと白身があり，その中にまた黄味がある，ゆで卵である，といってもよいでし

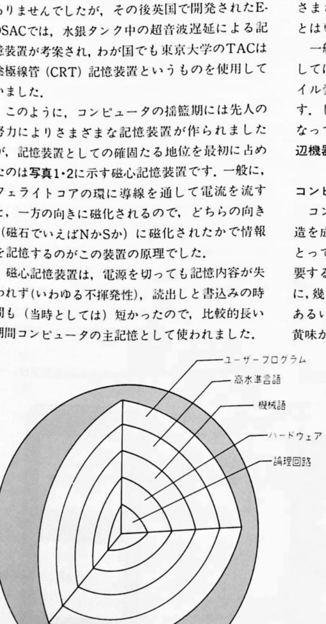


図1・2　コンピュータの階層構造

ょう．

大型コンピュータではもちろんのこと，マイコンでもハードウェアはきわめて大量のICで作られていますが，こうした回路（論理回路）の働きを一般利用者が知ることはありません．係わりのあるのは，設計者とごく限られた一部の人達だけです．したがって，これらは大多数の利用者にとって隠れたものとなっています．

この論理回路の外側には，ハードウェア層があります．そして，さらにその外側に機械語層があります．ハードウェアと一般にいう場合，この層を含めた内側を指します．

機械語層は，コンピュータを機械語で使用する場合のことで，実際にはアセンブリ語を使用して機械語命令の生成を行っています．しかしこれは一般の利用者にとって不便なので多くの場合インタープリタ（例：BASIC）で解釈したり，コンパイラ（COBOL，Pascal等）で機械語にして使用します．したがって，こうした高水準言語を操って，コンピュータを汎用データ処理装置として使用する人達には，この層も隠されたものとなります．

階層の一番外側はユーザープログラム層です．これはただ単に，できあがった応用プログラムを走らせ，結果を得るだけの層です．なお，ここにはまた，簡易言語による運用も含まれます．

簡易言語は，あらかじめ標準的なプログラムを多数用意し，これを組合せて一定の処理作業を行わせるもので，人間はルールに従ってデータを投入すればよいことになります．したがって，考えようによってはこれがもっとも人間に近い高度な水準となります．

このように，コンピュータの利用者は，運用の方法やその立場に応じて，コンピュータのさまざまな面を見ることになります．ある人が見ていることを別の人は見ていない，これがコンピュータの階層構造です．

1-1-3 マイコン，そしてパソコンの誕生

1971年，一つのLSIが誕生し，その年の暮れにわが国に紹介されました．PMOS技術という当時のLSI製造技術によって作られ，16ピンのセラミック容器に入れられたこのICの名称は4004といい，それまでの素子とは違ったものでした．これが世界最初のマイクロプロセッサです．処理の単位は，現在も制御用CPUがそうであるように，4ビットでした．

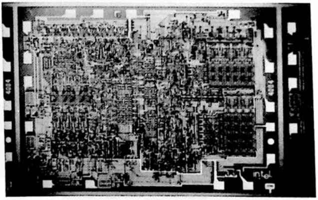


写真1・3　4ビットマイクロプロセッサ4004（Intel社提供）

4004の開発を担当したのは，米国のベンチャー企業インテルでしたが，発注したのは日本のビジコンという電卓メーカーでした．なお，この会社は不幸にして今はありません．

当時は，電卓の市場規模が拡大途上にある頃で，米国の半導体メーカーに電卓用LSIを発注していた各社は，各機種ごとに異なる機能を持った，いまでいうカスタムICの製造を依頼していたといいます．しかし，こうした事態は，LSIの製造者にとっても発注者にとっても好ましいことではなかったようです．メーカー側にしてみれば，カスタムLSIは需要が限定され，汎用製品のように広い市場性がありません．いわば魅力に乏しいわけです．一方ユーザーにとっては，開発期間が長びき，その分コストが高くなってしまいます．

マイコン誕生にまつわるエピソードを記した資料によれば，1969年ビジコンよりインテルに開発依頼があったのは，すべての電卓に共通して使えるもの，それを個別の製品にするにはプログラミング技術による，というものでした．これは，ハードウェアの機能の一部をソフトウェアで置きかえようとするもので，今日では非常に重要な技術手法と考えられています．まさに，コンピュータ制御そのものなのですから．

この4004マイクロプロセッサは，当時の技術資料とともにいまでも筆者の手元に残されています（写真1・3）．

4004の後，米国の周辺装置メーカからの依頼で，やはりインテルが開発したのが8008です．しかし，このマイクロプロセッサは本来の目的である端末装置には組込まれませんでした．端末装置の要求する仕様を満たさなかったのです．ただし，このLSIは世間一般が大いに注目するところとな

年代	4ビット	8ビット	16ビット	ビットスライス	その他
1971	4004（インテル）※	PPS－8△（ロックウェル）			
1972	PPS－4（ロックウェル） PPS－25（フェアチャイルド）	8008（インテル）			
1973	μPD751（日電）	8080°（インテル）	CP－1600（GI）		
1974	4040（インテル）※	MC6800●（モトローラ） F－8（フェアチャイルド）	MN1611（パナファコム）	3900（インテル）	TLCS12（東芝）
1975	TMS－1000（TI）	Z80°（ザイログ）		AM2900（AMD） SBP0400（TI）	
1976		8085°（インテル） MC6802●（モトローラ）			
1977		R6502△（ロックウェル）	TMS－9900（TI）		
1978			8086（インテル）		
1979		MC6809●（モトローラ）	Z8000（ザイログ） MC68000（モトローラ）		
1980			NS－16016（NS）		

（※，●，○，△などはそれぞれのファミリーの発展型を表わす）

表1・2　マイクロプロセッサの開発過程

参考文献

「3・4級入門」マイクロコンピュータの発生　(CQ出版社)

りました．一説によると，当時蜜に蟻がむらがるように顧客が殺到（？）したそうです．もちろん，筆者も買い求めました．8ビット汎用データ処理用マイクロプロセッサの源流は，ここにあるといってよいでしょう．

8008の市販開始は1973年です．この頃までは，LSIはPMOS技術により製造され，マイコン第1世代と区分されています．

パソコン時代へ

マイコン産業がはっきりとその基盤を確立したのは1974年頃からです．それには，前後して登場した8080，MC6800の力がありました．これらは8ビットマイクロプロセッサとして後年もっとも多くの人から関心を持たれたものです．

8080は，8008に対して上位互換性を有するよう設計され，8008のソフトウェアはすべて利用可能でした．以後この方針が貫かれ，8085からZ80に至る，80系と呼ばれるマイクロプロセッサの系列を作り出します．これに対し，MC6800は8080と競合する形で市場を獲得していきました．そして，いくつかの製品を経て，極め付けMC6809に至る68系を作り上げたのです．

1973～1977年をマイコン第2世代といいます．この世代の特色は，LSI製造技術として，PMOSに代るNMOS技術が確立したことです．**表1・2**は，マイクロプロセッサの開発過程を概観する年表です．

通信回線を使ったパソコン同士の対話

さて，1974 年発売されたALTAIRは，8080 の改良品 8080 Aを使用した組立てキットでした．そこで，これを皮切りに，今でいうボードコンピュータが数多く現われました．現在，パーソナルユースのコンピュータとしては，パソコンが全盛で，ボードコンピュータの用途は機器組込み用か学習キットですが，当時はこれがすべてでした．このため，ボードコンピュータが爆発的に売れたそうです．

組込み用マイコンには，通常プログラム作成能力がありません（言語処理系はもちろん，プログラムを自身の記憶装置に格納する機能を持っていない）．したがって，システム開発には開発ツールと呼ばれる支援機が必要で，これで作られた機械語プログラムが読出し専用メモリ（ROM）に書込まれ，目的の機器に組込まれるわけです．しかし，初期のボードコンピュータには，一種のオペレーティングシステム（モニタと呼ばれた）が準備されていたので，利用者はいろいろプログラムを走らせて楽しみました．だが，それもパソコンの登場まででした．

最初のパソコンは，1978 年に出たAPPLE IIです．CPUには 68 系の 6502 が使われました．APPLE社はこれによりパソコン市場における現在の地位を確立しました．シャープ株式会社からZ 80 搭載のMZ-80 Kが発売されたのもちょうどこの頃です．わが国ではパソコンユーザーがしだいに増加し，それにつれて新しい機種があいついで出ました．メーカー各社の新機種開発レースは，現在も激しく続けられています．

最近のパソコンの特徴としては，グラフィックス機能の著しい向上，サウンド機能の充実，漢字のサポートなどがあげられます．グラフィックスの機能は，コンピュータアートはもちろんのこと，ビジネス関係でも重要です．ただし，これにはディスプレイの性能も影響します．また，われわれ日本人にとって，仮名だけでなく漢字も使えないと困ります．

さらにもう一つの特徴として，通信インターフェイスの問題があります．初期のパソコンには，RS-232 C規格に準拠した直列インターフェイスを標準で備えたものは少かったのですが，近頃では，比較的低価格機でもこのインターフェイスを持っています（オプションのこともある）．ただし，筐体に刻まれた名称はRS-232 Cでも，実際は JIS C-6361 です．使用コネクタの形状がインチではなく，ミリ仕様である以上，RS-232 Cではあり得ないというわけです．

最近，この通信インターフェイスを介してパソコンを電話回線に結び，パソコン間通信を行う人達がふえてきました．日本マイコンクラブでも，パソコン通信による新しい会員サービスを開始しました．こうなると，異なる機種間での情報交換に関して，共通のプロトコルの決定，データ通信の信頼性など新しい問題も出てくるでしょう．今後のパソコン利用者には，ますます幅広い知識が必要になります．また，ソフトウェア関係者には，こうしたパソコンの高い能力を生かし，利用者のニーズに合致する質の高いソフトウェアの供給が望まれるわけです．

1-2 MZシリーズとMZ-1500

1-2-1 MZの誕生

電機メーカー各社がパソコン市場に参入し出したのは1979年頃からでした．それ以前からも工場の機械制御の一部を組み入れた機器などにパソコンを，いやパソコンというよりマイコンをとり入れたものが徐々に使われ始めてはいました．それを単独で使用できるよう，商品にしたものがいわゆるワンボードマイコンといわれるものです．

当初，このワンボードマイコンも何に使用するのか，何のためにこういうものがあるのかなどということが世間一般にはほとんど理解されておらず，一部のマニア，あるいは計測制御などを仕事に使用している人たちだけのものでした．16個ほどのキーから，アルファベットが羅列されたプログラムというものを無心に打ち込み，LEDが点灯したことだけでも当時大変な感動を覚えたことが思い出されます．

こういうワンボード形のマイコンから出発し，より身近な外見にして，何をするものかを少しでも分らせようとしたものがキーボードであり，またディスプレイ，データレコーダなどが組合わされてできたのがパソコンという商品です．

シャープでは，各社に先がけてこのパソコンをMZ-80Kという名前で1978年12月に発表しています．このMZ-80Kが商品として出される前にシャープもワンボードの「マイコンキット」MZ-40Kという商品（ROM，RAMやその他のICで構成し，ハンダづけを必要とした入門機）を発表しています．このMZ-40Kでは，センサ・オプション，鍵盤オプションなどを組合わせて，パソコンの機能，利用方法をマニア層に向けて説明を行い，将来，パソコン市場が拡大されるであろうと予想し，ユーザーを育成していました．それがMZ-80Kのデビューと同時に爆発的な人気を得，一体化のパソコンとしての国産第1号機として登場をしたのです．

ここで国産第1号機としてデビューをしたMZ-80Kの，簡単なスペックについて触れてみましょう．

まず，CPUボード，ディスプレイ，キーボード，データレコーダを一体化したことにより，コンパクトにまとまったことが普及した第1の理由でしょう．現在，ディスプレイや外部記憶装置などの多様化に伴い，一体化の商品がなくなりつつある中で，それぞれのオプションとの接続との手間などを，極力なくした特色を持つ商品でした．

また，それに加えて，商品を完成品として出さず，一部キーボード部分をキットとし，ユーザーの選択をあらかじめ商品側で行ったのです．購入者側から見れば，買えば誰でも使えるという感覚をなくし，ユーザーに対しある種の自信を植えつけさせたのです．

次に，では具体的な仕様についてを以下，述べてみます．

- MZ-80Kのおもな仕様
- CPU：Z-80（2MHz）
- ROM：4KB（モニタ用として使用）
- RAM：標準20KB，最大48KBまでボード内で拡張可能
- ディスプレイ：40桁×25行（1000文字表示）
- データレコーダ：シャープPWM方式を採用，ボーレイト1200bps（当時としては信頼性からも他メーカーの追従を許さなかった）
- ソフトウェア：BASIC言語を本体に付属した

そして，もう一つの大きな特徴であるクリーン思想というものが採用されています．これは将来いろいろなソフトが出てくるであろうことを予想し，そのソフトに対して最大限，有効なメモリエリアを確保し，使用ができるよう配慮した形となっています．

その後，80桁のドットプリンタ，5インチのフロッピーディスクなど，オプション機器の充実とともに成長していくことになります．その成長の過程において第1回目のモデルチェンジが行われました．1979年10月にデビューをしたMZ-80Cがそれです．MZ-80Kが第1号機であったこともあ

① MZ-80K

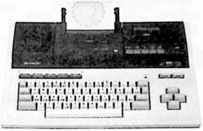
⇨

MZ-80C

⇨ MZ-80K2/K2E

MZ-1200 ⇨

MZ-700



⇨

MZ-1500

② MZ-80B

MZ-2000 ⇨

MZ-2200

③ PC-3100 → PC-3200 → PC-3500

④ MZ-5500

⇨

MZ-6500

シャープのパーソナルコンピュータの流れ

り，色あいをあまり重視していなかったことの対応，またその他の要望を集約した機種です．

同時に，MZ-80Kのソフトが100％使用できる完全コンパチブル設計をとったMZシリーズの大きな転機を迎えるのが，1981年4月に発表されたMZ-80Bです．これ以前に発表されたMZ-80K，Cの命令形態を継承し，かつ斬新なデザインで，当時パソコン業界をアッとおどろかせた商品です．この商品は約2年間のロングランを続け，後にMZ-2000/2200へと継承されていきます．

さて，ここでMZシリーズの流れについて触れてみましょう．

細かく分けると4つの流れとなりますが，④の16ビット機種については②，③のソフトウェアをすべて包含した形で形成されており，最終的な流れは2つとなり得るでしょう．

ここで注目すべきことは，昨今，ソフトウェアのコンパチビリティ（互換性）がさけばれる中で，1978年12月に発表されたMZ-80Kのソフトが5年以上も経った1984年5月に発表されたMZ-1500に使用できるということです．これは実に偉大なことであり，他のどのメーカーを見ても，ここまでの対応を施している機種は皆無といえます．

現在の市場においては新製品競争，つまりハード/ソフトの移り変わりが多く，各メーカーとも熾烈な競争の中で機能アップが要求されています．ややもすれば完全にコンパチビリティを失わせ，新しい流れを商品化したいのは当然のことといえるでしょう．それを踏まえつつ5年前のソフトが対応できる商品がMZ-1500であるということに注目して頂きたいと思います．

ここでMZ-1500の商品性について触れてみましょう．

MZ-1500は前に説明した通り，MZ-80K/Cの流れを汲みつつ，現在のパソコンで必要不可欠とされている機能をふんだんにとり入れられている機種です．それに加え注目すべき新しいメディアである，QD（クイックディスク）を最初に搭載した機種であることが最大の特徴といえます．

MZ-1500がいかにすばらしい機能をもっているかは後に触れるとして，ここでは企画段階の話に触れてみることにします．

MZ-80K/Cの後継機として1982年10月にMZ-700シリーズが市場に発表され，この機種も今までにない販売台数を記録しました．しかし，時間がたつにつれ，グラフィック機能の強化，あるいは音楽機能の強化，漢字処理への対応などユーザーの要求は日増しにつのるばかりです．

そういう中で，MZ-700シリーズの発売と同時に次機種の検討が進み，何とかユーザーの要望にこたえるものを，ということでスペックのつめが行われていきました．検討が進むにつれ当然MZ-700シリーズの機能強化ということだけではなく，新製品には何か他メーカーにはない目新しさが必要になります．いくつもの案が提出され，新機種に取り入れることが決定されました．

最終的に決定した中で，特に注目すべき機能がQD（クイックディスク）の標準搭載です．新製品を企画検討する際に，採用するものが他メーカーに先がけたものということになれば，特に試行錯誤の連続になります．事実このQDの採用までには技術検討の繰返しが行われ，簡単には実現できなかったのです．

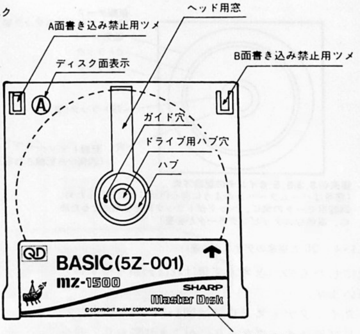
それはQDの容量は絶対に64KBなくてはいけない，という大前提で進行したからです．この技術検討のかいあって，最終的に1枚のメディア当り片面64KB，両面使用で128KBが実現できたわけです．

また，仕様の決定とともに，QD自体が全く新しいメディアであるための問題があります．

つまり現在，パソコンの外部記憶装置として一般に広く認知されているフロッピーやカセットと違うため，流通段階のつめも同時に行われていったのです．ここで問題となったのがメディアの価格です．QDは確かにカセットと比較すれば格段，いや場合によってはフロッピーと同等の速度で扱えるものです．しかし価格がべらぼうに高くては，QDを採用した意味がありません．価格は絶対にカセットテープなみか，それ以下にしなくては新しいメディアを採用した意味もないし，流通し一般に普及できないだろうということです．技術的検討が繰返され，その結果，最終的にブランクメディア1枚当り標準価格450円という価格が実現できました．

こうして，フロッピーの高速性とカセットの経済性の両面をそなえたクイックディスクが登場したのです．ただしメディアの容量の問題など，今後の課題が残されている事も事実といえます．

図1・3　クイックディスク



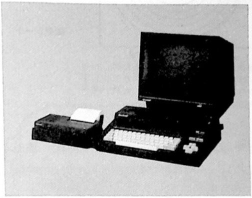


写真1・4　MZ-1500システム

1-2-2

MZ-1500の特徴

ここで，このクイックディスクを初めて搭載したMZ-1500の魅力をさぐってみましょう．

MZ-1500（写真 1・4）は，さきほどそのルーツを紹介したMZ-700シリーズの特長である

○家庭用テレビにつないですぐ使える手軽さ
○家庭用テレビ，CRTディスプレイ，ドットプリンタ，プロッタプリンタ，ジョイスティックなどの周辺機器を自由に選択できる柔軟性
○外部記憶装置などを本体に組込み，操作性を向上させたコンパクト設計
○メモリをオールRAMにし，自由にユーザー側で使用できるようにしたソフトウェアの柔軟性

などを基本にし，新たに

○カセットテープに比べて，データ，プログラムの入出力が速く，かつ操作性を重視したクイックディスクの標準装備
○いろいろな図形や絵が描けるグラフィック機能の強化
○多彩な音が作れる音楽機能の強化
○RAMファイル，ボイス機能や漢字処理などへの幅広い対応
○拡張基板用の内蔵スロットの装備

などの特長を追加したものです．

それでは，このMZ-1500についての特長をもう少し細かく紹介してみましょう．

1．クイックディスクの採用

このクイックディスク（図1・3）の特長は，何といってもアクセスが非常に速いことで，64KB（片面全部）を，なんと8秒で読書きすることが可能なのです．また，価格も450円とカセットテープなみの値段を実現しています．ただ，クイックディスクはフロッピーディスクのようなランダムアクセス機能がありません．〔ランダムアクセスとは，メディア上の任意の位置への読書きを行うことですが，図1・4に示すように，QDの構造上，トラック（データを記録する溝）が1本しかなく，レコード盤のようにうず巻き記録していくため，順番に読書きするシーケンシャルアクセスの方法となります．〕つまりQDはカセットテープをうず巻き

ディスク	2.8インチ
記録方式	MFM
記憶容量〔バイト〕	128K
記録密度〔BPI〕	4410
データ転送速度〔BPS〕	101K
トラック密度〔TPI〕	59
ヘッド数	1
トラック数	1
ディスク回転数〔RPM〕	423

表1・3　クイックディスクの仕様

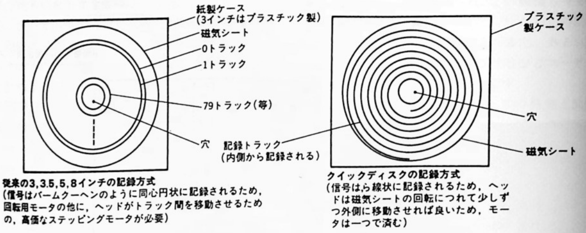


図1・4　QDと従来のディスクの違い

状にしたもの，と考えて頂ければわかりやすいと思います．

クイックディスク自体は図1・3に示すようにハードプラスチックケースにかこまれており，安全性も十分配慮された設計となっています．ディスクの仕様については，表1・3に示す通りであり，両面使用可能，記憶容量は128KBになります．反対側を使用する際にはカセットテープ同様，裏返して使用すればよいわけです．

○ドライブ用ハブ穴

ディスクドライブにセットしたとき，ドライブの回転軸がここに固定され，ガイド穴にガイドピンが固定されディスク本体を回転させます．

○ヘッド用窓

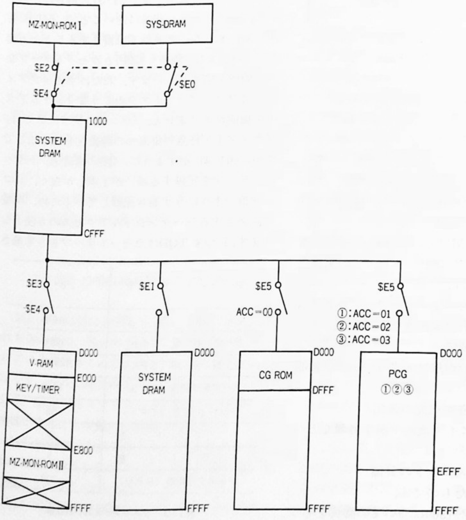


図1・5　メモリ構成

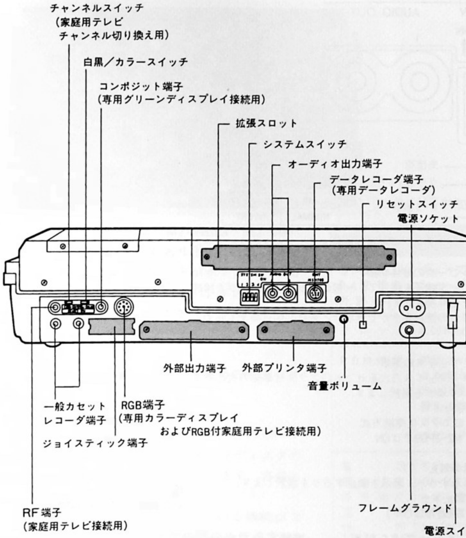


図1・6　各種入出力端子

読書き用ヘッドがこの窓よりディスク本体に接触して読書きが行われます．

注）ヘッド用窓の部分（磁気シート）には絶対に触れないで下さい．

2．グラフィック機能の充実

MZ-1500のグラフィック機能は，横320ドット×縦200ドットとなっており，そのすべてのドットに自由に8色の色指定が可能となっています．またPCG機能を含めて考えると，2つのモードをもっています．1つは320×200ドットのフルグラフィックス＋24個のPCG指定モード，もう1つは1000個のPCG指定モードです．当然PCGは，フォント（8×8ドット）の色指定が可能であり，ゲームなどの独自のパターンやスピードを要求されるものに十分対応できます．つまり，図1・5のメモリ構成を見て頂くとおわかりのように，メインRAMとのバンク切換えによって，使い分けがなされているわけです．しかし，これはすべてソフトウェアによる対応であり，PCGとグラフィックスの使い分けは可能です．

3．音楽機能の充実

PSG（プログラマブルサウンドジェネレータ）は，多彩な効果音や音楽の自動演奏ができる機能です．

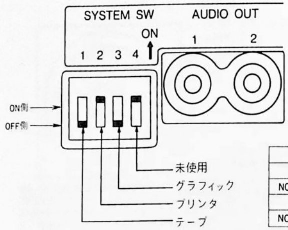
PSGは3和音＋ノイズのLSI（SN76489AN）を2個使用，2チャネルの出力端子をもっています（ただし，本体の内蔵スピーカー1個）．したがって，手持ちのスピーカー，アンプに接続すれば，ステレオ演奏も可能になり，バックグランドで音楽を楽しむことも可能になります．

4．各種出力端子の標準装備

図1・6を見て頂ければわかるように，豊富な出力端子群が装備されています．その中でまず

①ジョイスティック端子2個標準装備

MZ-1500のジョイスティック端子は，スイッチ



ON	OFF
—	—
NO DISP	DISP
MZ	セントロ
NORMAL	INVERT

本機の後面には，1～4までのシステムスイッチがあります．1～3のスイッチには次のような機能があります．

スイッチ1　：一般のカセットレコーダの極性切換
一般カセットレコーダ端子を使って，一般のカセットレコーダを接続した場合の極性を切り換えます．
カセットレコーダへの読み書きができないときは，切り換えてみてください．

スイッチ2　：プリンタインターフェイスの選択
プリンタのインターフェイス方式をセントロニクス社準拠方式にするか，MZ専用方式にするかを選択します．
ON　：MZ-専用方式
OFF：セントロニクス社準拠方式
※MZ-1P08，MZ-1P09ではON

スイッチ3　：グラフィック表示の禁止
グラフィックを表示するか，表示を禁止するかを選択します．
ON　：表示を禁止する
OFF：表示する(通常はOFF)

図1・7
レコーダとプリンタの切換え

式ジョイスティックと違い，アナログ式を採用しているために，どのくらい倒しているのかとか，どのくらい思い切って倒したかが正確にコンピュータに伝えられます．

②ディスプレイにマルチで対応

(1)家庭用のテレビに対応したRF端子．

(2)コンポジット入力（ビデオ入力）端子付きテレビ等に対応したビデオ出力端子．当然コンピュータ用グリーンディスプレイの接続も可能です．

(3) 8ピンRGB端子．このRGB対応ディスプレイに接続すれば(1)，(2)よりずっと鮮明な画像で見ることができます．

このクラスの機種では，この3つの内1つか2つしか標準装備していないものが多い中で（オプション対応は除く）MZ-1500では，出力端子を全て標準で装備しているのです．

③専用レコーダ(MZ-1T03)端子及び一般カセットレコーダ端子の装備

④外部出力端子

これはオプションの拡張ユニット(MZ-1U08)を接続するためのものです．

⑤外部プリンタ出力端子

MZシリーズには従来から採用しているMZ専用方式と一般的なセントロニクス方式の両方が接続できるよう対応しています．

③，⑤については，図1・7のように，本体後部のディップスイッチにより切換えを行っています．

5．各種インターフェイスの充実

MZ-1500は，4．で触れた標準装備の機能だけでも充実したマシンであるにもかかわらず，拡張性も十分に秘めています．まず，拡張スロットが2個標準で装備されています．この内1個は後述の汎用RAMファイル(MZ-1R18)を装着するためのスロットです．

もう1つはMZ-2000/2200シリーズと互換性のあるカードを装着できるものです．例えばフロッピーディスク（MZ-1F07）をそっくりそのまま使用できるほか，このスロットに差し込めるカードは他に漢字ROMカード（MZ-1R23)/（後述），RS

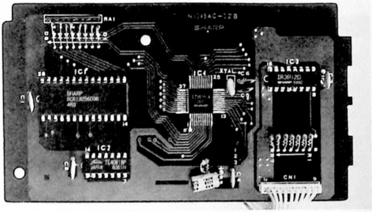


写真1・5　RAMファイル

-232Cインターフェイス（MZ-8BI03：MZ-700/2000/2200用と共用）などもあります．これに加えて，MZ-1500には拡張スロットがオプションで用意されており（MZ-1U08），これは，本体後部の拡張I/Oバスに接続することでさらに２枚のカードを取付けることが可能となります．

６．MZ-1500の機能を引きだすための強力なオプション群

①RAMファイル（MZ-1R18）

このカードを写真1・5に示します．このカードは種々の用途に利用可能で，クイックディスクの機能を最大限引き出せる武器といえます．容量としては64KBです．

現在のBASICでサポートされている機能は２つあり，１つはRAMファイル（外部記憶装置）としての機能，もう１つはプリンタスプーラとしての機能です．

これらは一方だけの機能を使用することも，また適当な容量（BASICにて指定可能）で区切って使用することもできます．もちろんRAMファイルはフロッピーディスクよりデータの受渡しは高速ですし，外部記憶装置として２台目のQDのように使用することもできます．しかも，クイックディスクの内容をそのままコピーしたり，逆にRAMファイルの内容をそのままクイックディスクにコピーすることもBASICのコマンド（LOADALL，SAVEALL）で簡単に行えます．

もう１つのプリンタスプーラとしての機能について説明します．これは簡単にいえばプリンタバッファのことです．つまり，パソコンでプリンタを使用する（例えばプログラムのリストを印字する，プログラム中でプリンタを動かす）場合，プリンタの速度（印字速度）はパソコンがプリンタに対してデータを送出する速度よりもかなり遅く，膨大なプログラムなどを印字させる場合，終了するまで待っていなくてはいけません．

そこでこの不便さを解消するために，プリンタに出力したいデータを一度RAMファイルの方にまとめて転送してしまい，プリンタが１字または１行分動作を終えた時に，逆にプリンタからデータのリクエストをもらうのです．その時だけ次の

方　　式	正弦波重畳方式
音声合成LSI ＤＡ変換器 外部ＲＯＭ	シャープL 130951A 内蔵RPM 4 KB シャープ2 R 3 R12D 7 bit D A 32KB
クロック	4.19MHz
ＲＯＭ内容	英語66WORDS 日本語196WORDS 休止18種類 擬音18種類 メロディー7種類 スピードコントロール3種類
電　　源 使用温度 保存温度 湿　　度 外　　形	5 V ±0.5V 0 ～60℃ −20～70℃ 使用湿度85％以下 140（W）×79（D）＝基板面積

表1・4　MZ-1M08の仕様

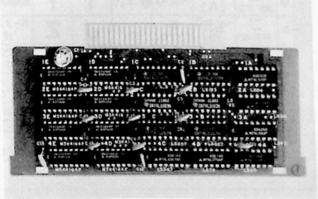


写真1・6　ボイスボード

①内部ROM

		上位4bit			
		0	1	2	3
下位4bit	0	Oh	Ten	Zero	チャイム1
	1	One	Eleven	Time is up	チャイム2
	2	Two	Twelve	to	チャイム3
	3	Three	Thirteen	It's	チャイム4
	4	Four	Fourteen	It's now	休256ms
	5	Five	Fifteen	go to	休 16ms
	6	Six	Sixteen	ぶ～ん	休192ms
	7	Seven	Seventeen	ピッ	
	8	Eight	Eighteen	ピッ ピッ	
	9	Nine	Nineteen	ブッ	
	A	Hours	A.M.	ブーン	
	B	Hour	P.M.	ぴっ	
	C	Minutes	Twenty	ピッピッピッピッ	
	D	Minute	Thirty	Alarm	
	E	休 8ms	Forty	Alarm set for	
	F	休 8ms	Fifty	(メロディ)	

②外部ROM

		上位4 BIT															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
下位4 BIT	0	ゼロ ゼロフンティカに使用	ハッ	SP 128ms	0 ゼロ	キュ^ウ	P	よくできました	たす	Dティ	～経過	ピャ	時	タ	ミ	ピャ	デ
	1	分フン	ジュウ～(A)	休 8ms	1 イチ	A	Q	笑い声	ひく	～億	ニャ	ピュ	ア	チ	ム	ピュ	ド
	2	分プン	ジュウ～(C)	休 16ms	2 ^ニ	B	R	ゲームスタート	イテテッ	キャ	ニュ	ピョ	イ	ツ	メ	ピョ	パ
	3	休止大 200ms	ジュッ(テン)	休 32ms	3 サ^ン	C	S	ゲームオーバー	イテッ!	キュ	ニョ	リャ	ウ	テ	モ	ガ	ピ
	4	休止中 80ms	～ジュッ(テン)	休 48ms	4 ^ヨン	D	T	アウト	自動車(全)	キョ	午前	リュ	エ	ト	ヤ	ギ	プ
	5	休止小 40ms	ジュウ	休 64ms	5 ^ゴ	E	U	GO	自動車(発進)	ニ (時計'時'用)	午後	リョ	オ	ナ	ユ	グ	ペ
	6	ブッ	～ジュウ(A)	休 160ms	6 ロク	F	V	命中	自動車(走行)	ロッ (時計用)	ヒャ	ヲ	カ	ニ	ヨ	ゲ	ポ
	7	ブ～ン	～ジュウ(B)	休 256ms	7 ^ナナ	G	W	お好きなKEYを押して下さい	自動車(エンディング)	シャ	ヒュ	ギャ	キ	ヌ	ラ	ゴ	バ
	8	ピッ	ヒャク	休 320ms	8 ハチ	H	X	YかNのキーを押して下さい	メロディ 軍艦行進曲	シュ	ヒョ	ギュ	ク	ネ	リ	ザ	ビ
	9	ニ (時計'分'用)	～ヒャク(A)	～です	9 ^キュウ	I	Y	最高得点	メロディ アキシーランド	ショ	ヨ(時)	ギョ	ケ	ノ	ル	ジ	ブ
	A	ゴ (時計用)	～ヒャク(B)	× (カケル)	ニ	J	Z	ボーナス点	メロディ 螢の光	シエ	ワ(時)	ティ	コ	ハ	レ	ズ	ベ
	B	ゴ (50分用)	～ヒャク	+ (プラス)	^サン	K	ゴー	あなたの番です	メロディ おおスザンナ	ニー	ミャ	ファ	サ	ヒ	ロ	ゼ	ボ
	C	～ジュッ (時計用)	～ヒャク	E	ヨ^ン	L	まちがいです	私の勝です	メロディ 故郷の人々	チャ	ミュ	ジャ	シ	フ	ワ	ソ	SLOW
	D	CR	～セン	− (マイナス)	= (イコール)	M	正解です	文法上の誤りです	メロディ 村のかじや	チュ	ミョ	ジュ	ス	ヘ	ン	ダ	
	E	イッ	～ゼン	● (テン)	ゴ	N	O.K	私はMZです	休止 480ms	チョ	～時間 2,4,5,9ご使用	ジョ	セ	ホ	休止1 120ms	ヂ	MIDD
	F	ロッ	～マン	/ (ワル)	ナ^ナ	O	もう1回チャレンジしますか	点	休止 340ms	チエ	～時間 1,3,6,7,8ご使用	ジエ	ソ	マ	休止2 90ms	ヅ	FAST

表1・5　ボイスボードの内容

1字，または1行分のデータをプリンタに転送してやるわけです．そうすればプリンタとパソコンの並行処理が可能となり，その余った時間でさっそく次のデータ処理が可能となり，全体的に処理速度の向上が図られることになります．

②ボイスボード（MZ-1M08）

このボイスボードを**写真1･6**に示します．ボイスボードとは，コンピュータが言葉をしゃべるようにすることができるものです．数字，カタカナ，アルファベットのデータを音声として出力したり，特定の音声データを出力したりするものです．**表1･4**にその仕様をまとめておきます．

また，このボイスボードでは，外部ROMに示すように34個の固定メッセージがあらかじめ登録されており，BASICで簡単に出力することができ，プログラム中のいろいろなメッセージにも応用できます(**表1･5**参照)．

③プリンタ

MZ-1500は，MZシリーズの（MZ専用の）各種プリンタと，一般的なセントロニクス方式の両方のプリンタを接続できます．また4色カラープロッタプリンタも準備されており，BASICのコマンドで簡単に円を描いたり，作図をさせたりすることができます．プロッタは一般的にスピードが遅いのが弱点ですが，RAMファイルのスプール機能を使用することにより並列処理が可能となりますので，ほとんど気にならない状態になります．

④漢字ROMボード（MZ-1R23）

この漢字ROMボードを**写真1･7**に示します．MZ-1500には，約3,500字もの漢字が表示できる漢字ROMボードが取りつけられます．この中には記号，英・数字，ひらがな，カタカナ，ギリシャ文字，ロシア文字などの非漢字も含まれており，常用する文字としては十分に日常生活に対応できるものです．それぞれのドットパターンは16×16

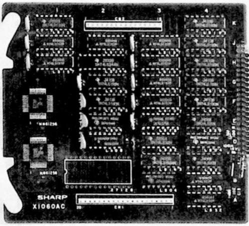


写真1・7　漢字ROMボード

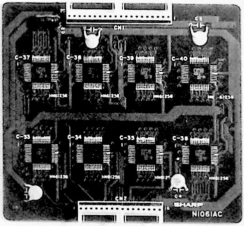
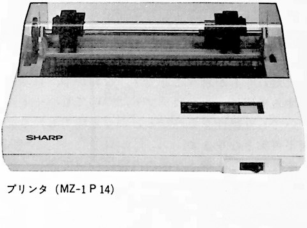


写真1・8　辞書ROMボード

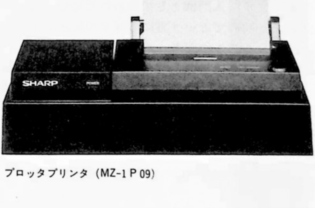
ドットで形成されています．また，1文字単位でしか表示できない不便な命令ではなく，漢字列を扱えるような命令もBASICで対応しており，操作性は，8ビット機中最高のクラスに入るといえます．

⑤辞書ROMボード（MZ-1R24）

このクラスというより16ビット機種を含めたパ



プリンタ（MZ-1P14）



プロッタプリンタ（MZ-1P09）

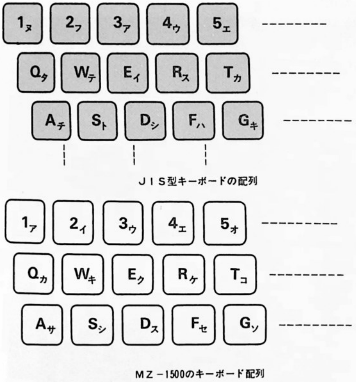


図1・8
JISとMZ-1500の
キーボード配列

ソコンの中で漢字辞書(写真1・8)をハードで対応させているのはMZ-5500/6500以外ではこのMZ-1500しかなく，もちろん8ビット機中では，初めてです．登録辞書数は約8万語であり，日本語ワープロとして，また，その他事務処理などに十分すぎる程の機能をもったものです．

以上がMZ-1500の簡単な特長ですが，内部のもっと詳しい解説については後にゆずるとして，ここでは特に豊富なオプション群について注目してください．

次に，MZ-1500がどういう形で使われていくかということです．各種の注目すべき機能を標準で装備し，かつ新メディア(QD)の標準搭載など，話題にこと欠かないハード構成，ソフト構成となっています．また，価格的に10万円を切ったこと，現在の市場の中では価格的には入門機クラス，機能的には中級機クラスに十分入り込める仕様となっています．入門機として使われた場合，特にパソコンは初めてという方にも種々の配慮が施されており，その1つがキーボードといえます．

現在一般的に普及しているのはJIS型のキーボード配列です．ところが，MZ-1500のキーボードは多少変わっていて，カナが50音順になっているのです．これについては図1・8を参考にしてください．

このような配列は，パソコンをはじめて使用する人にとってとり組みやすく，キーボードアレルギーの1つの大きな原因となっているカナ入力を簡素化しています．また，使用頻度の高い，いくつかのキー(＋，－，＊，(，)など)がSHIFTキーとの併用でなく，ワンタッチで入力可能になっています．この他にも，例えばディスプレイの選択，プリンタの選択，前機種との互換性，外部記憶装置の選択やその他周辺機器，ボード関係の対応など，種々の面で入門機にとどまらず，マニア層にも十分満足できる設計となっています．

ソフトウェアについても，MZ-1500の周辺機器を応用し，発展性が大いに秘められています．例えば，通信関係のソフトに対してもRS-232Cの対応，漢字・辞書ROMボードを活用した中級機なみの事務処理等への応用，また，Z80アセンブラ他，システム開発用言語ツールの対応などクリーンコンピュータのメリットを大いに活用し，1つの目的にとどまらず，多方面への活用ができる構成となっています．ユーザーの立場からいえば，当然すぐれた機能のものを，できるだけ安くということはだれしもが思うことです．なおかつ，将来に対する発展性(これはソフトについてもハードについても互換性をもたせるという意味でも)も必要不可欠となります．

それでは，次章よりその内容について詳しく述べていきます．

第2章 パソコンの基礎

2-1：パソコンの構成

中央処理装置
入力装置
出力装置
入出力装置

2-2：ソフトウェアの役割

ソフトウェアの歴史
ソフトウェアの概念
IPL
モニタ
BASICインタープリタ

2-3：LSI素子について

Z80CPU
Z80PIO
Z80PIOのアーキテクチャ
Z80PIOのプログラミング
Z80SIO
8255
8253
CRT, メモリコントローラ
PSG

2-1 パソコンの構成

パソコン（パーソナルコンピュータ）は，大型コンピュータと比較すると非常にコンパクトで，一見非力なように見えますが，その構成や計算させる方法（アーキテクチャと呼びます）については本質的に変りありません．

パソコンは大型コンピュータほどメモリの容量や演算速度に余裕がなく，またプログラム作成の際に使用できる命令の種類も多くありませんが，価格が非常に安く，だれでも身近に置いて手軽に使用できるという長所を持っています．

パソコンに限らず，どのコンピュータでも同じように，大きく3つの部分から成り立っています．これらはそれぞれ**中央処理装置，入力装置，出力装置**と呼ばれています（図2・1参照）．

中央処理装置とは，ＣＰＵやＭＰＵとも呼ばれ，その名の通りパソコンの中心となる最も重要な部分です．ここにはプログラムやデータを記憶する部分があり，プログラムの内容に従って，記憶や演算，他の装置の制御といった処理を行なっています．

入力装置とは，パソコンにプログラムやデータを与えるための装置です．

現在最も多く利用されているのはキーボードです．このほかマウスやライトペン，デジタイザといったものもあり，最近ではオンラインシステムのような通信回線を介して各種のデータ入力を行なう本格的な装置もパソコンに相当普及してきました．

最後に出力装置ですが，パソコンでさまざまな処理を行なっても，処理した結果を出力しなければ，何の役にも立ちません．結果を人間にわかる形に表示させるのが出力装置です．

代表的なものにディスプレイ装置やプリンタ，X-Yプロッタなどがあります．

コンピュータは，入力装置からプログラムやデータを入れ，これを中央処理装置で処理し，その結果を出力装置で外部に表示するというのが，基本的な構成です．

先に述べました通り，中央処理装置には記憶装置がありますが，多量のデータなどを処理する場合には，この記憶装置だけでは容量不足になることもあります．こういった場合のために，主記憶装置以外に補助記憶装置というものがあります．

主な使い方としては，プログラムやデータの保存，再生をすることで，パソコン用にはカセットテープ，フロッピーディスク，クイックディスク，ハードディスクなどがあります．つまりパソコンの外部記憶装置といえるわけです．

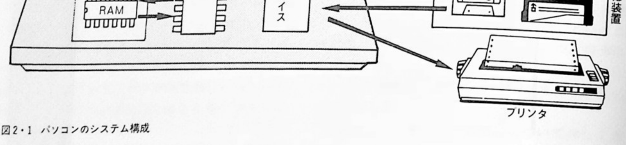


図2・1　パソコンのシステム構成

この外部記憶装置も含めて考えると，パソコンは，4つの部分から成立っているといえるでしょう．

2-1-1
中央処理装置

(1)CPU

パソコンの中心部であり，制御と演算の機能を1つのチップにしたLSIのことで，これをCPU（Central Processing Unit，中央処理装置），またはMPU（Microprocessor）と呼びます．

中央処理装置という言葉は，大型コンピュータの呼び方にしたがったものなので，マイクロコンピュータのCPUを特にマイクロプロセッサ（MPU）と呼ぶこともあるのです．

CPUは数千～数万個のトランジスタで作った回路と同じ機能を，わずか数mm角のLSIに収めた大規模集積回路です（写真2・1）．このCPUによって，メモリの中にある命令が順次実行されて行きます．

コンピュータの内部では，すべて0か1で処理されます．つまり2進法の世界です．

これには電流が流れているかいないか，磁気があるかないかなど，2つの状態を0と1に対応させて，その組合せで数値や各種命令を生み出しているのです．

この0と1の最小単位をビット（bit）といい，それが並列処理できる情報量によって，4ビットCPU，8ビットCPU，16ビットCPUなどに分かれます．

4ビットCPUでは，$2^4=16$の情報を区別できるので，0～9の数は表現できますが，アルファベット26文字のすべてを表現することはできません．8ビットCPUでは$2^8=256$種の情報を区別できますので，数字やアルファベットだけでなく，カナ文字まで表現することが可能です．

現在はこの8ビットCPUが主流ですが，この8ビットを1つの単位としてバイト（byte）と呼びます．メモリの単位もこのバイトで表わされています．

英文字やカタカナなどの文字を示す場合，通常1文字を1バイトで表わします．例えば「CPU」という文字情報をパソコンのメモリに記憶させる場合，3バイトが必要になるわけです．

私たちが日常使っている言葉は数も多く，かつ大変複雑なので，コンピュータに直接理解させることは，到底できません．

そこでCPUに演算や制御をさせる命令は，最終的に0と1の2進数で表現されます．

この0と1の数で表現される命令の集まりを機械語といいます．機械語は，使用するCPUに特有の命令であり，CPUの種類によって機械語も異なります．

パソコンに処理を行なわせるための言語（各種命令の集まり）も，いろいろ開発されていますが，直接CPUと会話ができる機械語が最も高速に処理できるのはいうまでもありません．

CPUに動作をさせるためには，外部から正確なクロック信号を与えてやらなければなりません．つまりCPUの各動作は，このクロック信号で制御されます．

例えば，フラッシュをたいて写真を撮るときのことを考えてみて下さい．フラッシュは一瞬強い光を出すだけですから，シャッターとフラッシュのタイミングが少しでもずれると，写真はうまく写りません．それと同じように，ひとつの情報をCPUからメモリへ送るタイミングと，メモリが情報を受け入れる状態になるタイミングが，ピッタリ合っていなければなりません．そのため正確な間隔でパルス（電圧の波）を発生させるのがクロックなのです．マイクロコンピュータの各部はクロックからのパルスが高圧になる一瞬一瞬に合わせて動いているので，CPUが情報を出すのと，メモリが体勢を整えるのが同時にでき，全体のタイミングが合うわけです．

(2)記憶装置（メモリ）

記憶装置は，パソコンに必要な情報を記憶する装置で，CPUと同様に頭脳に相当するところで

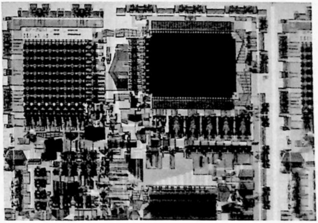


写真2・1　LSIの拡大写真

す．メモリの中には，データを書込んだり，変更したり，消去したりできるものと，いったんデータを書込むと変更できないものの，２種類に分かれます．

このデータを変更できるメモリをＲＡＭ（ランダムアクセスメモリ），書込みや変更のできない，読出し専用のメモリをＲＯＭ（リードオンリーメモリ）と呼びます．

パソコンのプログラムやデータなどは，何度も入力や出力をするので，当然 RAM を使用しますが，**モニタプログラム**（パソコンにプログラムを入力するなど，基本的な制御用プログラム）などは読出しだけでよいので，ROM を使用します．

このように，パソコンのメモリには，RAM と ROM が使用されています．

2-1-2 入力装置

パソコンに仕事をやらせようとしても，プログラムとデータがなければ何もしてくれません．このプログラムやデータをパソコンに与えてやるのが，入力装置です．

最も簡単なものはキーボードで，超大型コンピュータからホームコンピュータまで，コンピュータと呼ばれるものにはまず付属されています．

最近ではマウスやタブレット，ライトペンなども普及してきており，将来的にはキーボードを使わなくても，パーソナルコンピュータを活用できる日がくるでしょう．

(1)キーボード

キーボードには大別して，ASCII キーボードとＪＩＳキーボードの２種類があります．ＡＳＣＩＩキーボードは英文字を基本にしたもの，ＪＩＳキーボードは英文字，カナ文字の両方を使う目的のものに用いられています．

入力できるのは，英大文字（普通のタイプライタと異なり，大文字が基本になっています）や数字のほかに，シフトキーを押せば小文字，カナキーを押せばカナ文字を打つことができます．またグラフィックのための記号キーも備えており，文字以外に，キー入力で図形を描くこともできます（図 2•2)．

キーを押すことで８個の０または１の信号が出るようにしておくと，256 種までの文字や記号を打つことができるわけです．

つまり８桁の２進数のどれがどの文字に対応するかをあらかじめ決めておけばよく，このような約束ごとのひとつにＡＳＣＩＩコード（アスキーと読みます．American Standard Code for Information Interchange の略）があります．（図 2•3 参照）

例えば英文字の「A」は 16 進数で“41”，10 進数では“65”となりますが，パソコンの内部では“０１０００００１”となっているのです．

(2)その他の入力装置

最近よく用いられるものにマウスやライトペン，ジョイスティックなどがありますが，これらはキーボードと異なり，座標入力装置といえます．

つまりこれらの装置を動かすことによってディスプレイ上の座標を移動させ，実務ソフトやゲームをする上で，キーボードを使わずに済むことを目的としたものです．

この他，人間が手で文字などを書いて直接パソ

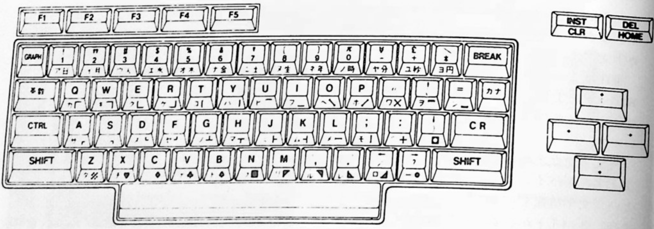


図 2・2　MZ-1500 のキーボード

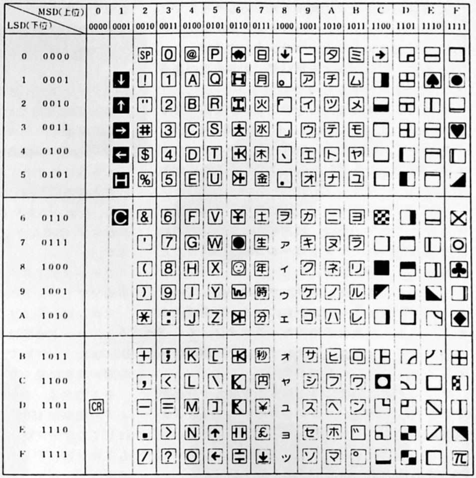


図2・3
ASCIIコード表

コンに理解させる手書き入力装置や，人間のしゃべる声を理解させる音声認識装置なども実用段階に入っており，今後ますます便利な入力装置が開発されてゆくでしょう．

そうなると，本当の意味でコンピュータと対話できる日がくるかもしれませんね．

2-1-3 出力装置

パソコンが実行し，処理した結果を人間が分かるように表現させるのが出力装置です．

出力装置の代表的なものとしては，ディスプレイやプリンタがあります．

ディスプレイは，パソコンの実行している処理を即座に表示してくれるので，大変便利な出力装置といえますが，記録を残すことができない欠点があります．そこで，記録を保存する必要がある場合に用いるのがプリンタです．ディスプレイ上では，一度に表示できる情報量が限られていますので，長いプログラムリストなどを見る時にも便利といえます．

(1)ディスプレイ

パソコン本体から画像情報を受けとり，画面に表示するのがディスプレイです．

パソコンでプログラムを作成していると，自分が入力した文字や命令などが，ディスプレイを通してパソコンに入っていくような錯覚を覚えることがありますが，全く逆なわけです．

パソコンとディスプレイなどが一体型となったオールインワンタイプと呼ばれるものもありますが，最近ではパソコンと別に置いて使用するセパレートタイプのものがほとんどのようです．

パソコンによっては，家庭用のテレビと接続して使用できるタイプがありますが，文字だけでなく絵や図形などのグラフィックも表示する必要がありますので，できればパソコン専用ディスプレイで，しかも解像度の高いものを選ぶ方がよいでしょう．

(2)プリンタ

大型コンピュータで使用されているラインプリンタなどは，パソコンではほとんど使用されておらず，**ドットマトリクスプリンタ**が広く用いられています．これは活字を用いず，文字を点の集合としてプリントするものです．

感熱式または放電式などのプリンタもありますが，これらは特殊な紙又はリボンを必要とします．

最近では主に図形を印字させるプロッタプリンタや，カラーで非常に美しいグラフィックを印字できるカラープリンタも普及しています．

パソコンからプリンタへの印字は，文字情報だけでなく，文字の形状，紙送り量の設定などの処理が，定められたコントロールコードを送ることによって行なわれます．

プリンタの文字は，プリンタ自身が持っているキャラクタジェネレータから，パターンを読込みます．つまり画面表示用のキャラクタジェネレータ内にある文字でも，プリンタ内になければ印字することはできないわけです．

プリンタ選択の際にはこれらの考慮も十分必要ですから，やはりパソコン本体専用のプリンタをお薦めします．

(3)その他の出力装置

ユニークな出力装置としてパソコンに言葉をしゃべらせる**音声ボード**があります．プログラムの作り方次第で，パソコンと会話させることも可能です．

その他，パソコンと連動して，精密な図形を画かせるための**X-Yプロッタ**や，キーボードとプリンタが一体となったテレタイプ(TTY)などがあります．

2-1-4 入出力装置

メモリのRAMが記憶している内容は，パソコンの電源を切るとすべて消えてしまいます．またその容量にしても，多量のデータ処理を必要とするプログラムでは，収まりきらないことがよくあります．

これらのことから，**主記憶装置**（CPUが直接制御するメモリ）をカバーする**補助記憶装置**が必要になります．

最も広く用いられているのは**カセットテープ**（音楽用で結構です）で，大容量や高速性を必要とされる場合には，**フロッピーディスク**や，**クイックディスク**，**ハードディスク**などがあります．

(1)クイックディスク（QD）

MZ-1500ではじめてパソコンに搭載された新しい記憶装置で，フロッピーディスクの高速性とカセットテープの経済性を備えています．

大きさは約2.8インチでカセットテープのように裏表使用でき，両面で128 KBの容量をもっています．片面，つまり64 KBのロードは約8秒しかかからず，価格もカセットテープなみの1枚450円です．

クイックディスクの良さを端的にいうならば，市販ソフトウェアの利用でしょう．

フロッピーディスク版のゲームを楽しもうと思うと，ロード時間は速いものの価格が高くなります．またテープ版では価格が安い代わりにロード時間が非常に長くなります．

そこでクイックディスクなら，もともとのメディアの価格が450円ですから，ゲームパッケージにしてもテープ版とほとんど変わりません．しかもフロッピーディスクなみのスピードがあるので，即座にゲームソフトをロードし，楽しめるというわけです．

このようにメリットの多いクイックディスクは，今後ますます流通が盛んになるでしょう．

(2)フロッピーディスク

フロッピーディスクは，ポリエステルの表面に磁気コーティングした円盤で，四角いジャケット（保護ケース）に入っています．

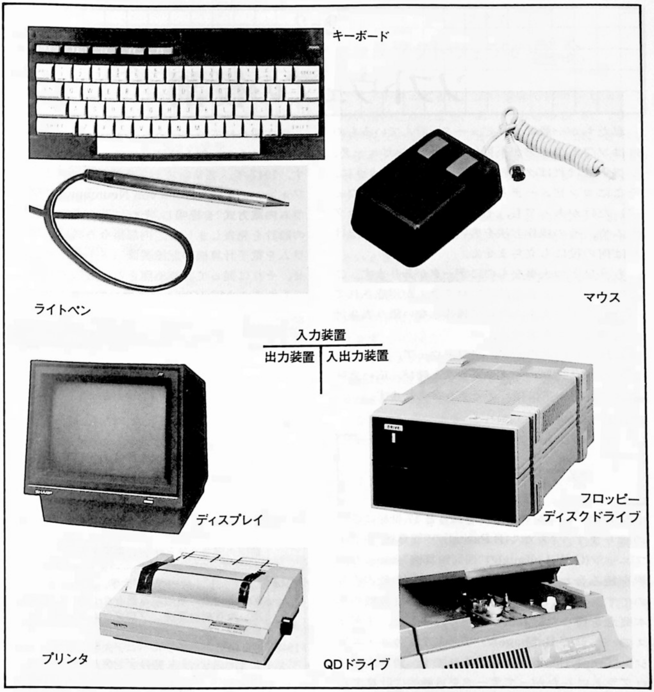
種類は円盤の大きさで区別されており，8インチのものを標準フロッピー，5.25インチのものをミニフロッピーと呼び，3.5インチや3インチのマイクロフロッピーやコンパクトフロッピーもあります．

(3)ハードディスク

ハードディスクは，フロッピーディスクと違い硬質なディスクを用いているため，ディスクの回転速度もフロッピーディスクの10倍程度速く，記憶容量も大きくなっています．

(4)その他の入出力装置

MZ-1500には，RAMファイルという，入出力



装置が用意されています.

RAMファイルとは,その名の通りメインメモリとは別にRAMエリアを設け,そのRAMをフロッピーディスクのようなファイルと同じように扱えるものです.

フロッピーディスクは読み書き速度が速いのですが,媒体をもちいているため,アクセス(読み出しや書込み)を行なうのに,必ず待ち時間が必要になります.

しかしRAMファイルなら,CPUとの電気的な信号のやりとりですむわけですから,ほとんど待ち時間が必要ありません.

使い方としては,RAMファイル上でプログラムやデータ作成を行い,完成した結果をクイックディスクに登録するというスピーディな処理ができます.

また,RAMファイルの一部のエリアをプリンタバッファ(プリンタへ送るデータの容れ物のようなもの)としても使用できますので,プリンタに印字しながら,画面では全く別の処理を行なうというユニークな処理も可能です.(この機能をプリンタスプールモードと呼びます.)

2-2 ソフトウェアの役割

私たちが一般にコンピュータと呼んでいるものにはソフトウェアが不可欠です．「コンピュータ，ソフトなければタダの箱」といわれているように，そこにコンピュータと呼ばれる機械（ハードウェア）だけがあっても，それを動かすためのプログラムや，その操作方法を書いた操作説明書がなければ何の役にも立ちません．

もうひとつ大事なものにデータがあります．ここにパソコンと表集計のプログラムが用意されていても，データを入力して操作しない限り表集計の結果はでてきません．

これらハードウェア，ソフトウェア，データをまとめてコンピュータの三要素と呼び，広い意味のソフトウェアには，データも含まれます．

2-2-1 ソフトウェアの歴史

現在のパソコンのルーツを探ると17世紀にさかのぼります．パスカル(B.Pascal)の“加算機”やライプニッツ(G.W.Leibniz)の“四則演算機”といった歯車を組み合わせた機械が計算の機械化の最古のものです．そして1830年頃，現在の電子計算機の基本概念を持った計算機が考案されました．イギリスのバベジ(C.Babbage)の“アナリティカル・エンジン”(analytical engine)がそれです．歯車式でプログラムにしたがってデータを自動的に計算するものでした．このとき，バベジの計算機のためにプログラムを作成したのがオーガスタ・エイダ・ラブレース夫人(Augusta Ada Lovelace)です．この女性の名前はAda（エイダ）というプログラミング言語の名称にも使われています．

世界初の電子計算機とされている“ENIAC”が1946年，ペンシルバニア大学で完成します．18,000本以上もの真空管からなるこの計算機は，まだ現在のコンピュータにはほど遠く，プログラムごとに配線盤のプラグを差替えなければなりませんでした．

現在使われているほとんどすべてのコンピュータは，ノイマン型と呼ばれているコンピュータです．1947年，プリンストン高級研究所のジョン・フォン・ノイマン(John von Neumann)は**“プログラム内蔵方式”**を提唱し，それに基づく電子計算機の設計を発表しました．内部指令方式は，プログラムを電子計算機の記憶装置（メモリ）に記憶させ，それに従って計算処理を行うものです．

もちろんMZ-1500もノイマン型のコンピュータ

年代	おもな内容
1642	パスカルの計算機（歯車式，加減算のみ）
1694	ライプニッツの計算機（歯車式，四則演算ができる．ライプニッツは2進法の創始者といわれている）
1834	アナリティカルエンジン（チャールス・バベッジ／ケンブリッジ大学）　機械式，パンチカードにより演算法と数値を与える．入力装置，記憶装置，演算装置，制御装置を備えたものだったが，設計図のみでついに完成しなかった
1890	パンチカード式統計作成表機（ハーマン・ホレリス）1890年の米国国勢調査に使用された．前回の調査に比べ半分の期間で完了した
1944	Mark　I（ハーバード大学，IBM） 3000個以上の電磁継電器とたくさんの歯車を用いた自動計算機．プログラムはスイッチ，制御盤，紙テープでなされた
1946	ENIAC（ペンシルバニア大学，エッカートとモークリー）　初のデジタル電子計算機．18,000本の真空管，7,000個の抵抗，1,000個のコンデンサ，6,000個のスイッチ，1,500個のリレーが使用されている．プログラムはスイッチまたはプラグの差しかえにより与える
1947	フォンノイマンがプログラム内蔵方式提唱
1949	EDSAC（ケンブリッジ大学，ウィルス） プログラム内蔵方式を採用して最初に完成した電子計算機
1950	EDVAC(ペンシルバニア大学ムーア電気工学教室，モークリーとエッカートなど）　プログラム内蔵方式を最初に採用した電子計算機(1944年開発開始)．ただしEDSACよりも完成が遅れた
1951	UNIVAC-I（Eckert and Mauchly Co.）初の商用電子計算機

表2・1　コンピュータ第1世代の年表

年代	プログラミング言語の概要
1956	FORTRAN I IBM704のためにバッカスが開発
1957	FLOWMATIC(COBOL言語の原型) UNIVACⅠ／Ⅱ用米海軍のホッパー女史
1958	ALGOL58 国際算法言語委員会が発表
1960	COBOL データ・システム言語協議会が開発
1963	BASIC ダートマス大学のクメニーとクルツ
1966	PL／I SHARE・IBM
1968	Pascal チューリッヒ工科大学　ヴィルト
1981	Ada言語 米国防総省が開発

表2・2　おもなプログラム言語の年表

なのです．

その後，世界初の“プログラム内蔵方式”の計算機の“EDSAC(エドザック)”が1949年に，ノイマン型の“EDVAC（エドバック）”が1950年に，それぞれ，ケンブリッジ大学と，ペンシルバニア大学で完成されました．

ENIACの時代から商用の計算機（ノイマン型）が誕生した時代（～1957年）までを第1世代と呼びます（表2・1）．

第2世代（1958年～1963年）は多くの言語が開発されました．その代表的なものがFORTRAN（1956年）COBOL（1957年）などです（表2・2）．

OSもコンピュータの発展につれて形成されました．初期のものは非常に簡単なものだったといわれます．初期のOSとして有名なものにノースアメリカン航空会社のものがあります．このOSは，操作員の介入を減らすことで，システムの稼動率を向上させコンピュータに連続で作業させようというものでした．

この後，時代は第3世代（1964年～1970年）に入り，集積回路の使用によりコンピュータは小型化，高性能化してゆきました．そして，大型コンピュータを多数の独立の使用者が共同利用可能とするタイムシェアリングシステム(TSS)，本格的なOSが開発されました．

そして，現在までを第3.5世代（1971年～）と呼んでいます．LSIやICメモリの開発，発達が進み現在にいたっています．

2-2-2
ソフトウェアの概念

ソフトウェアとは，広い意味ではコンピュータの利用のしかたなど，ありとあらゆる“形”であらわされないものを指しますが，ここでは，オペレーティングシステム(Operating System 略してOS)やプログラミング言語，アプリケーションプログラムのことを指します．

それぞれの関係は図2・4のようになっておりソフトウェアはハードウェアと利用者の中間にあるといえます．

一般にOSの制御の下で，プログラミング言語やアプリケーションソフトが動き，アプリケーションソフトは，プログラミング言語によって開発されます．

パソコンの下位機種や中級機では，このOSとBASICの区別がはっきりしていないことが多く，電源をONにするとすぐにBASICの入力待ち状態になってしまいます．これは，OSと本来のBASIC言語がBASICの中に混在しているためです．

（1）オペレーティングシステム(OS)

コンピュータを効率よく使うための管理プログラムで，コンピュータの規模に合わせて様々な規模のものがあります．パソコン用の主なOSを表2・3に載せておきます．

最も規模の小さいOSはモニタです．モニタもOSだというと違和感を覚える方がいるかもしれませ

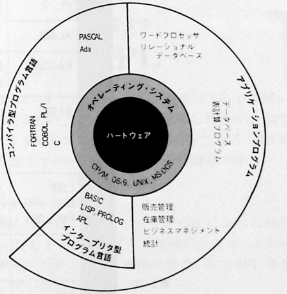


図2・4　ハードウェア，OS，言語，アプリケーションソフト

んが，プログラムの呼び出しやキーボードやCRTディスプレイなど入出力装置の制御（管理）をしていることに他なりません．

OSの目的をもう少しくわしく述べますとシステムの効率，操作性，拡張性，柔軟性，稼動率の向上です．

大型コンピュータでは稼動率を上げるためにマルチタスクという方法がとられており，1台のコンピュータを複数台の端末から同時に複数の人間が利用できるようになっています．複数のプログラムを時分割により，同時並行処理できるのが，大型コンピュータのOSでは当りまえです．

一方1回に1本のプログラムしか動かせないパソコンでは，OS……モニタの仕組みは簡単になります．

ところで大型コンピュータのマルチタスクほどではありませんが，MZ-1500のBASICには，プリンタスプール機能というものがあります．オプションのRAMファイルを取付けた時にのみ可能なのですが，プリンタに文字を打っている間にも他の処理（例えばグラフィック）ができるというものです．そしてもう1つ，音楽関係の命令も他の処理と並行して行なうことができます．

OSの目的は，今まで述べたもの以外にもう1つあります．それは互換性の向上です．

AというOSの制御の下で動くBASICやFORTRANなら，AというOSが動く別のパソコンでもこれらが利用できるわけです．といっても，そのままでは利用できない場合もあるのですが，一部を変更するだけで良いので，移植性がたいへんに高いといえます．

表2・3のパソコン用のOSのほとんどは入出力制御，ディスクの管理機能，テキストエディタやアセンブラ，各種言語を統括しており，一部のも

	CPU	オペレーティング・システム
8ビット	8080 Z-80	CP/M　MP/M UCSD－P system
8ビット	6809	FLEX, OS－9
16ビット	8086 (8088)	CP/M－86, MS－DOS
16ビット	68000	CP/M－68K, UNIX

表2・3
パソコン用のおもなOS

低レベル言語	機械語	コンピュータが直接実行できる．一般に2進数列であり、プログラムの作成がめんどうでまちがいやすく誤りの修正がむずかしい
低レベル言語	アセンブラ	機械語を人間の理解しやすい記号（ニモニック）で表せるようにしたもの
高レベル言語	FORTRAN	初の高レベル言語．科学技術計算用．ライブラリの蓄積が威大
高レベル言語	COBOL	事務処理用言語、処理手続き、データ、ハードウェア環境を分離して記述する．データが階層構造をもつ
高レベル言語	PL／I	科学技術計算用と事務処理用を総合した汎用処理言語
高レベル言語	ALGOL	アルゴリズム記述用言語
高レベル言語	PASCAL	教育用言語．ALGOL、および構造化プログラミングの影響を強く受けている
高レベル言語	BASIC	ほとんどのパーソナルコンピュータで利用できる．初心者向けに、学びやすく使いやすいことを目的に開発された．しかし、現在パーソナル・コンピュータに採用されているものは、この目的にそっているとはいいがたい
高レベル言語	LISP	人工知能用に開発された非数値計算用
高レベル言語	PROLOG	述語論理型言語．自然言語に近い記述ができる
高レベル言語	C	システム記述用言語．構造化プログラミングを支援する．表記が単純．アセンブラに近いこまかなプログラムを書ける
高レベル言語	Ada	米国国防総省が組み込み形計算機システムのために制定

表2・4
プログラミング言語

のは，マルチタスク機能も有しています．

（2）プログラミング言語

パソコンには必ずといっていいほどBASICが用意されています．では，BASICとはなんなのでしょうか．

コンピュータはハードウェアだけでは動かないことを前に述べました．コンピュータを利用するためには，ソフトウェアが必要です．具体的にいえば，コンピュータに対して，その動作（計算）の手順を示したもの（手順書）やデータがあって初めて利用者の考えている動作（計算）をさせることができるのです．この，動作（計算）の手順を示したものこそがプログラムであり，プログラムを書き表わすための人工の言語がプログラミング言語なのです．

また，BASIC言語のように，人間にとってわかりやすい，数式と簡単な英文などを使ってプログラムを記述する言語を，**高水準言語**といいます．

BASIC言語で書いたプログラムを実行するためのプログラムがBASICです．

プログラミング言語には，表2・4のようなものがあります．

プログラミング言語は大きく分けて，インタープリタ型とコンパイラ型の2つの方式があります．これ以外に，プログラムと機械語が1対1に対応するアセンブラ言語もあります．

インタープリタ型の言語にはBASIC，LISPなどがあげられ，人工の言語で書かれたプログラム（ソースプログラム）をインタープリタ自身が翻訳しながら実行する形式のものです．

コンパイラ型の言語には，FORTRAN, C, Pascalなどがあります．人工の言語で書かれたプログラム（ソース・プログラム）をコンピュータが直接実行できる機械語のプログラム（オブジェクトプログラム）に翻訳してしまってから実行する形式がコンパイラ型の言語です．一度で機械語に翻訳させてしまうことをコンパイルといいます．

インタープリタ型の言語とコンパイラ型の言語のそれぞれの特長は，インタープリタ型の言語は，ソースプログラムを翻訳しながら実行するのに対して，コンパイラ型の場合は，一度コンパイルしてしまえば，後は翻訳の必要がないので実行が非常に高速です．

その反面，コンパイラ型言語は図2・5のような

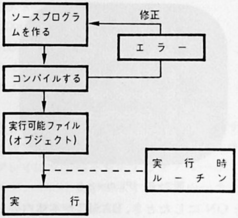


図2・5　コンパイラ言語のプログラム作成手順

手順でプログラムを作成，修正しなければならないのに対し，インタープリタ型の場合は，すぐにプログラムの作成，実行，修正ができる手軽さがあります．

先にインタープリタ型の言語の例にBASIC，コンパイラ型の言語の例にPascalを挙げましたが，コンパイラ型のBASIC，インタープリタ型のPascalも開発されており，言語によって型がきまっているのではないことを記しておきます．

（3）アプリケーションプログラム

アプリケーションプログラムとは，コンピュータの最終利用者が使用するプログラムです．例をあげればきりがありませんが，ビジネス用ソフトと呼ばれている物や，ゲームなどがこの層に入ります．

アプリケーションプログラムは，先に述べたプログラミング言語を使って作成するのが一般的です．

2-2-3 IPL

IPL（アイピーエル）という言葉は「あまり聞いたことがない」という方が多いかもしれません．IPLはイニシャルプログラムローダの略で，MZ-1500のようなクリーンコンピュータの必需品です．

（1）IPLの働き

MZ-1500は，電源スイッチがONになった瞬間からIPLというプログラムを実行します．ふだん何気なく行なっている操作の中で，実はIPLを使用していたのです．

BASICが記録されたQDをセットして電源スイ

図2・6　IPLのメッセージ

ッチをONにしたとき，BASICが本体内のRAMに読込まれるのは，実はIPLの働きによるものなのです．

つまりIPLは，電源スイッチをONにしたりリセットスイッチが押されたときに，QDやフロッピーディスク，カセットからBASICなどのプログラミング言語を読込むのが仕事なのです．

QDやフロッピーディスクをセットせずにスイッチをONすると図2・6のような画面が表われます．これはBASICなどのプログラムを，どの装置から読込む（ロードする）のかをたずねているメッセージで，それぞれの装置名の頭文字をキーボードから入力してやることで指定します．

この中で“M：Monitor”というメッセージがありますが，このときキーボードの“M”を押すとROMモニタへ移り，ROMモニタのコマンド（命令）待ちになります．MZ-1500には，このROMモニタと，後述するRAM上のBASICモニタがあります．

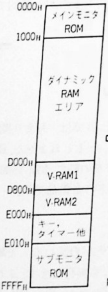


図2・7　電源ON時のメモリマップ

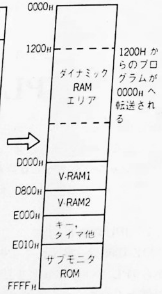


図2・8　BASICをロードしたとき

（2）IPLの構造

BASICのQDをセットして電源スイッチをONするとBASICが起動されますが，以下のような仕組みでBASICが起動されるのです．

a）　電源スイッチがONになった時，メモリマップは図2・7のようになっています

b）　8255，8253，PSGなどの周辺LSIを初期化します

c）　VRAMをクリアします

d）　PCGをクリア（初期化）するためのプログラム48バイトをC000Hに転送しC000Hにジャンプします

e）　バンク切換えをしながらPCGを初期化します

f）　サブモニタのPIO初期化ルーチンへジャンプします

g）　SIOを初期化します

h）　キーボードのキーが押されていないかチェックします（かりに押されてなかったことにします）

i）　QDから一番先頭のファイルのインフォメーションブロックを読込みます．インフォメーションブロックには，ファイルの種類，ローディングアドレス，サイズ，実行アドレスなどが記録されています

j）　ファイル名を画面に表示し，続いてBASIC本体を1200Hからロードします

k）　メモリマップを図2・8の状態にし，1200HからのプログラムをG0000Hへ転送後，0000Hへジャンプします

これでBASICが起動します．

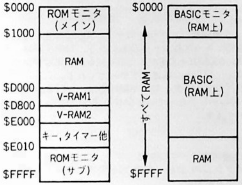
h）　でキーが押されていて，プログラムをロードする装置がFDやカセットになった場合もほぼ同じ手順で起動されます．

キーが押されていなくてもフロッピーディスクのI／Fが取り付けられていれば，フロッピーディスクからロードします．何かエラーが起きなければ，図2・6の入力待ち状態の画面になります．

2-2-4 モニタ

（1）モニタの働き

モニタは，「ソフトウェアの概要」で述べたよ



ROMモニタのメモリマップ　BASICモニタのメモリマップ

状　　態	モニタ
●Ⓜを押しながらRESETスイッチを押した場合	ROMモニタ
●Ⓜを押しながら，電源をONにした場合	ROMモニタ
●起動時にモニタを指定した場合	ROMモニタ
●BASIC上でBYE命令を実行した場合	BASICモニタ

図2・9　ROMモニタとRAMモニタ

うに，オペレーティングシステムの一種であり，BASICなどのプログラミング言語の手助けを行ないます．具体的にはクイックディスクや，カセットテープなどの制御やファイルの管理，ディスプレイやキーボードの入出力用サブルーチンプログラムの集まりで，機械語プログラムの作成などの機能をもっています．

MZ-1500のモニタは2種類あります．ROMモニタとBASICモニタです．ROMモニタはIPLから実行を移すことができ，キーボードの“M”を押しながらリセットスイッチを押すことでもROMモニタに実行を移せます．

BASICモニタは，IPLから起動したBASICの中に含まれています（図2・9参照）．BASICのコマンド入力待ち状態でBYE CR とキーから入力すると，BASICモニタの入力待ち状態に移ります．

モニタのコマンド表を表2・5に載せておきます．

（2）モニタの構造

BASICモニタやROMモニタは，どのような構造になっているのでしょうか．OSの目的は前にも述べましたが，システムやソフトウェアの開発の効率を良くするために，モニタは，入出力装置をコントロールするためのサブルーチンを多くもっています．サブルーチンの一部をあげますと

a）各種LSIの初期化
b）キーボードからの文字入力
c）ディスプレイへの文字表示
d）BREAKキーのチェック
e）QDへのデータセーブ
f）QDからのデータロード
g）MUSICの処理
h）タイマのセット
i）モニタコマンドの入力と実行

のようなものがあげられますが，サブルーチンはさらに複数のサブルーチンからなっています．ディスプレイに文字列を表示するサブルーチンは，ディスプレイに1文字表示するサブルーチンをくり返しているといえます．

モニタがサブルーチンプログラムの集まりであることは，これでおわかりいただけたでしょう．もうひとつ，モニタには重要な機能があります．それは，モニタコマンドの処理です．モニタのコマンドには，メモリに直接，機械語を書込んだり，実行するコマンドや，メモリ上に作成したプログラムやデータをQDやカセットにセーブ，ロードするコマンドがあります（表2・5参照）．コマンドが入力されるとそれぞれのコマンドの処理ルーチンから必要なサブルーチンを呼ぶようになっています．

2-2-5 BASICインタープリタ

BASICはプログラミング言語の項で述べたように，インタープリタ型の高水準言語です．

BASICは，私たちが普段使用している言葉に近い簡単な英語(PRINT，INPUT他)と，数式などでプログラムを作成することができます．そして，BASICは，私たちが作成したプログラムを翻訳しながら実行するわけです．ここでは，BASICインタープリタの実行の仕組みについて説明します．

（1）BASICインタープリタの構成

MZ-1500のBASICの中にはモニタが含まれています．そして，モニタは各種のサブルーチンの集まりです．BASICインタープリタも同様多くのサブルーチンから構成されています．図2・10はBASICモニタとBASICインタープリタの機能をブロック別に分けて書いたものです．それぞれの機能＝サブルーチンプログラムになって考えていただいて結構です．

これらの機能は，さらに細かく分けることもできます．また，図中に書かれていない機能もあり

コマンド	9Z-502M(ROMモニタ)	BASICモニタ
*J0000H [CR]	指定アドレスにジャンプ	*G 0000H [CR] ただしサブルーチンコール
*L [CR]	カセットよりロード後実行	実行アドレスの指定 同左　ただしSAVE時があるときのみ実行
*F [CR]	FDをブート (EB39Hへジャンプ)	メモリサーチ (指定範囲で数列をサーチする)
*B [CR]	ベルON/OFF	××××
*# [CR]	メモリチェンジ (バンク切替え)	××××
*PABC [CR]	文字列をプリンタへ出力	メモリダンプなどがすべてプリンタへ出力される
*M0000 [CR]	メモリ書換え	同左
*S [CR]	カセットに機械語プログラムをセーブ	同左 (さらにコロンに続けてファイルネームも指定する必要あり)
*V [CR]	カセット上の機械語プログラムをベリファイ	同左
*D0000 [CR]	メモリの内容をダンプする	メモリの内容をダンプする

9 Z-502M…ROMモニタ（サブ）
1 Z-009B…ROMモニタ（メイン）

*QL [CR]
　QDより機械語ファイルをロード．ファイル名を指定しないと，最初にみつけたファイルをロード．ロード開始，実行番地が共に0000Hだと自動的に1200Hからロードしてメモリバンク切替え，ブロック転送までやってくれる．
　また，ファイル長，ロード開始番地，実行番地のバッファーが1104Hからとなっていて CMTの場合（1102H）とずれていますので注意．

*QS [CR]
　QDに機械語ファイルをセーブ．これにより，すべての機械語プログラムIPL起動（電源ONと同時にロード，実行する）とすることが可能．

*QF [CR]
　すべての新品のメディアは1度フォーマットをかけてから（裏・表）使用する必要がある．また，フォーマットをかけるとそれまでのファイルはすべて消えてしまう．BASIC（MZ-5Z-001）でINIT"QD:" [CR] でフォーマット可能．

*QX [CR]
　カセットよりファイル（任意のモード）をロードしそれをQDに移す．これにより，700用のHuBASIC, EDASなどが簡単に乗る．

*QC [CR]
　QDからQDへファイルコピー．これも任意モードのファイル（OBJ, BTX, BSD），で可能．
　修正してからコピーしたい場合は，途中でBREAKして修正後FIFIHへジャンプさせる．なお，必ず1200Hからロードされるので実際のロード開始番地と異なる場合は注意が必要．

*QD [CR]
　メディア上のファイルのディレクトリを調べる．ファイルの属性（OBJ, BTX, BSD, BRD）のあとに*がついているのはそのファイルがロックされている，つまりDELETEできないことを表すが，もともとQDはフォーマットをかけない限りファイルのDELETEは可能．
　これはMZ-1500全般についていることだが小文字，ひいてはコントロールコードの扱いがROMモニタ，BASICMZ-5Z000, HuBASICですべて異なる．このためBASIC上とモニタ上で見掛け上ファイルネームが変わってしまい，ロード，コピーができなくなる恐れがある．（工夫することで可能．）しかしこれを逆に利用してファイルのプロテクトに使うという手もあります．キーボードから直接入力できない文字をファイルネームにすればよい．たとえば，BASIC上でSAVE CHR$( ) [CR] などとすればほかの人はロードできなくなる．

*EB [CR]
　CMOS-RAMボードをブート

*ES [CR]
　CMOS-RAMボードに，カセットから読込んだプログラムをセーブ．

表2・5　モニタコマンド

ます．

（2）プログラムの記憶のされ方

BASICが起動して最初にすることは"初期化"です．その後，BASICのコマンド待ちになります．

ここで

10　A＝1 [CR]

と入力したとき，メモリにはどのように記憶されるのでしょうか．

10　A＝1 [CR]

と入力した後，

BYE [CR]

でBASICモニタに移ってください．そして，

DA 215 [CR]

と入力すると，A215Hから，メモリにどのような状態で記憶されているかを，見ることができます．

BASICインタープリタは，

10　A＝1 [CR]

とキーボードからプログラムが入力されるとA215番地から図2・11のように変換されたデータ(16進数）が入ります．

BASICでは，行番号の後に命令を書くというきまりがあります．最初の2バイトはその行が何バ

図2・10 BASICモニタとインタープリタの機能別ブロック図

イトの長さなのかを示しています．0D00は上位と下位が逆になっており，本来の値は000DHで，10進数に変換すると13です．つまりこの長さは13バイトということです．

次の2バイトもやはり上位と下位が逆になっていますが行番号を表わしています．

次のスペース（空白）は，メモリ上には記憶しません．

次は変数名のAです．これはAのアスキーコードである41（16進数）が入っています．

＝は中間コードというものに変換されて記憶されています．中間コードについては後述します．

次の6バイトは数値の1を，浮動小数点内部表現という形に変換して記憶したものです．

最後の7バイトは，現在メモリに記憶されているプログラムの一番最後の部分であることを示しています．プログラムに続きの行があるときにはこの部分は変わります．例えば

10　A＝1

の続きに

20　B＝3

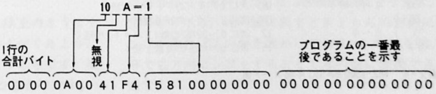
が入力された場合7バイトのうちの先頭の2バイトが0D00Hに変わります．0D00Hは，次の行へ移ることを意味する“改行コード”です．0D00Hの後へは，次の行の長さ，行番号………と続きます．

a．中間コード

中間コードとは，FOR　GOTO　PRINT などの予約語を圧縮（コード化）したもののことです．表2・6は中間コードを一覧表にしたもので，BASICの予約語は，この表に従って変換されメモリに記憶されます．この変換の際に生じた異常は Syntax Error となります．

b．浮動小数点内部表現

図2・11
プログラムとその処理の例



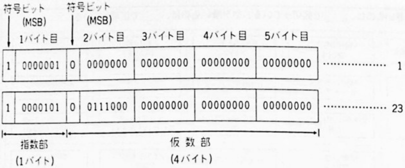


図2・12
浮動小数点内部表現の構造

表2・6の中で「15　次に5バイトの実数が続く」があります．最初の例のA＝1の1は実数で，それを変換したものが　158100000000　であることは説明しました．1バイト目の15は中間コードです．その後に続く5バイトはどのように変換されたのでしょうか．

コンピュータは，内部の演算を2進数で行います．10進数の　1　を2進数に変換すると　1　で，23ならば　10111　になります．

MZ-1500のBASICでは，5バイトのうち1バイト目を指数部，後の4バイトを仮数部とし，正規化という作業をへて，浮動小数点内部表現に変換しています（図2・12）．

ただし，仮数部の符号ビットは，後で説明する変数テーブルに記憶するときにしか使いません．仮数部で負の数を表わすときには中間コード15の前に，F8を記憶することで表現します．この方が速く演算ができるからです．

変換の過程を23（10進数）で説明します．

10進数の23は，2進数では　10111　で，正規化すると 0.10111×2^5 です．2^5の指数の5は，10111を左へ5ビットシフトしたことを表わしています．参考までに 10111 は，

$$1011.1 \times 2^1$$
$$101.11 \times 2^2$$
$$10.111 \times 2^3$$

L/H	8x	9x	Ax	Bx	Cx	Dx	Ex	Fx	FE+8x	FE+9x	FE+Ax	FF+8x	FF+9x	FF+Ax	FF+Bx	FF+Cx
x0	GOTO		POKE	HSET		ROPEN	TO	=<		COLOR		INT		CHR$		
x1	GOSUB	INPUT	OFF	GPRINT	ERROR	XOPEN	STEP	<=	CSET	PRTY		ABS		STR$		
x2			PMODE	KEY	ELSE		THEN	=>	CRESET	SET	MUSIC	SIN		HEX$	HEXCHR$	
x3	RUN	IF	PSKIP	AXIS	USR		USING	>=	CCOLOR	RESET	TEMPO	COS			ERN	
x4	RETURN	DATA	PLOT	LOAD	BYE		n	=		LINE	CURSOR	TAN			ERL	TI$
x5	RESTORE	READ	PLINE	SAVE		DIR	ALL	>		BLINE	VERIFY	LN			SIZE	POINT
x6	RESUME	DIM	PLINE	MERGE			TAB	<		PAL	CLR	EXP			CSRH	EOF
x7	LIST	REM	MOVE	CHAIN	DEF		SPC	+		CIRCLE	LIMIT	SQR			CSRV	FN
x8		END	RMOV	CONSOLE		RENAME		−		BOX		RND		SPACE$	POSH	
x9	DELETE	STOP	TRON	SEARCH		KILL		¥		PAINT		PEEK			POSV	
xA	RENUM	CONT	TROFF	OUT@	LABEL	LOCK	XOR	MOD	SOUND	POSITION		ATN		ASCCHR$	LEFT$	
xB	AUTO	CLS	INP@	PCIRCLE		UNLOCK	OR	/		PATTERN		SGN		ASC	RIGHT$	
xC	EDIT		DEFAULT	TEST		INIT	AND	I	NOISE	HCOPY	BOOT	LOG		LEN	MID$	
xD	FOR	ON	GET	PAGE			NOT	∧	BOOP	KPATTERN		FRAC		VAL	FONT$	
xE	NEXT	LET	PCOLOR	WAIT	WOPEN		><	(Ext.)	VOICE	FPRINT		PAI	JOY			
xF	PRINT	NEW	PHOME	SWAP	CLOSE		<>	(Fun.)				RAD				

数値に対する中間コード

00　1行の終了
01 ～ 0A　0～9
0B　次に2バイトの行番号が続く
0C　次に2バイトのアドレスが続く
0F　次に2バイトの16進が続く
12　次に0～32767の数（2バイト）が続く
15　次に5バイトの実数が続く

表2・6　MZ-1500 BASIC中間コード表

1. 単純変数

1	1	n	3又は5 (バイト)
型	名前長さn	変数名	データ

型——数値変数 05H
　　文字変数 03H
データ——数値変数 5バイトFAC (実際の数値)
　　　　文字変数 3バイト（最初の1バイト——文字列長 (0～FFH)
　　　　　　　　　　　　　次の2バイト——文字列エリアの先頭からのオフセット

(例) A＝1.5を実行すると，変数テーブルは

05	01	"A"	81	40	00	00	00

(例) AB$＝"MZ" を実行すると，変数テーブルは

03	02	"AB"	02	offset

文字列エリア
offsetバイト
"MZ"

2. 配列変数

1	2	1	n	1	2		2	(5又は3)×全要素数
型	全バイト数 m	名前長さn	配列名	次元数 ℓ	第1次元添字	………	第ℓ次元添字	データの並び

（全バイト数以降）mバイト

型——数値配列 85H
　　文字配列 83H
データの並び——単純変数の場合の5バイト又は3バイトのデータが
　　　　　　　配列の要素の数だけ並んでいる.

(例) DIM A (3) : A (1) ＝0 : A (2) ＝1.5 : A (3) ＝－2.5を実行すると

85	0016	01	"A"	01	0003	00	00	00	00	00	81	40	00	00	00	82	A0	00	00	00

3. 関数 (DEF FN)

1	2	1	n	m－(n＋3)	
型	全バイト数 m	名前長さn	関数名	中間コードになった関数定義	00

（全バイト数以降）mバイト

型——数値関数 45H
　　文字関数 43H

(例) DEF FNA$ (B$) ＝LEFT$ (B$, 1)

43	0016	01	"A"	"(B$)"	F4	FF BA	"(B$,"	15	81	00	00	00	00	")"	00

F4: ＝の中間コード
FF BA: LEFT$の中間コード
15: 5バイト実数であることを示す中間コード

(注) 変数テーブルは単純変数，配列，関数，あるいは数値の文字の区別なく連続して置かれており，型が00になったところで変数テーブルが終っている
変数テーブルの先頭は1074H番地からの2バイトに置かれている
文字列エリアの先頭は1076H番地からの2バイトに置かれている

図2・13
MZ-1500の変数テーブル

$$1.0111\times 2^{4}$$

$$0.10111\times 2^{5}$$

のように変換できます．正規化とは上のように小数点第1位には必らず最上位桁の1がくるようにし，最上位桁の0.1を省いてしまうことです．従って仮数部は 011100000…… となるわけです．

指数部は，2^5の5を2進数7桁で表現したものに符号ビットを付けたした 10000101 になります．符号ビットは，元の値に戻すときどちらの方向にシフトすればよいかを表わしており，1で左方向，0で右方向にシフトすることで元の値に変換できます．

浮動小数点内部表現で表わされる最大値は

15 FF 7F FF FF FF

で最小値は

F8 15 FF 7F FF FF FF

となります．最初のF8は符号の“－”を意味する中間コードです．

もっとも0に近い値は，正では

15 01 00 00 00 00

負では

F8 15 01 00 00 00 00

になります．

c．変数の管理

BASICで扱える変数の種類は文字変数，数値変数，配列変数（文字，数値）があります．配列変数は，DIM命令の宣言によって文字配列変数，数値配列変数をそれぞれ1次元から4次元まで扱うことができます．

この他に，これらと同様の管理をされるものとして，ユーザー定義の関数があります．

DEF FNA$(B$)=LEFT$(B$, 1)

などがそれです．

BASICはプログラム実行の際に，これらの変数が現われるとメモリ上にある変数格納エリアに，図2•13のような値を入れ記憶していきます．そしてA=B+CやPPINT B$などのように変数の値を参照しなければならない場合には，これらをすばやく読み出します．この変数格納エリアのことを変数テーブルといいます．

変数テーブルの先頭アドレスは，1074番地（16進数）からの2バイトを読み出すことでわかります．1074番地が3Dで1075番地がA2ならば，A23D番地から変数テーブルがあることになります．

図中の単純変数とはA=1.5，B$=“ABC”などの数値変数や文字変数のことで，それぞれ管理のしかたが違います．数値変数については，図のデータの部分が5バイトの浮動小数点内部表現になっています．文字変数の場合にはデータの部分に直接，文字列が入っているのではなく文字列の長さと記憶されている場所が入っています．データの部分は3バイトからなっており，1バイト目は変数に代入された文字列の長さ（00～FFの16進数）が入っています．2バイト目と3バイト目には，文字列エリアの先頭から何番目に記憶されているかを示すデータが入っています．

文字列エリアとは文字変数に代入された文字列が入っている場所で，1076番地（16進数）からの2バイトに，その先頭のアドレスが記憶されています．

文字列エリアの先頭アドレスがA23E番地で図2•13中の例のoffset（何番目に記憶されているか）が0020（16進数）だったとするとA23E番地に20を加えたA25E番地から2文字が文字変数AB$に代入される値ということになります．

配列変数についても同様で図中のデータの並びは数値配列変数の場合は，5バイトの浮動小数点内部表現がそのまま入り文字配列変数の場合は，文字列の長さと記憶されている場所を示す値の合計3バイトが入ります．

関数については型や関数名に続く関数の定義が中間コードで記憶され，関数呼び出し時にはその中間コードにもとづいて処理されます．

MZ-1500の変数名は，すべて2文字まで有効となっていて，3文字以上の変数名を使用しても変数名テーブルには，先頭の2文字しか記憶されません．この制限を解除する方法があるので紹介します．

BASICのPOKE文などで8263H番地をF1Hに書きかえると変数名の長さは255文字まで識別可能になります．

POKE $8263, $F1 [CR]

また，78BDH番地を1AHに書き換えると，2文字目以降に予約語を含んでもよいようになります．例えば

APRINT, MTOP

なども許されるわけです．

POKE $78BD, $1A [CR]

この2行をBASICでAUTO RUNするようにしておけば，大変便利です．

変数の管理で注意しなければならないのは変数テーブルには，プログラムの入力時に書込まれるのではなく，プログラムの実行時（ダイレクトモード）にインタープリタが1つ変数を見つけるたびに，変数テーブルに書込んで行く点です．

10 A=1.5

とプログラムを入力した段階では，変数テーブルには何も書込まれません．

（3）プログラム実行の方法

a．ダイレクトモード

行番号を付けずに命令（コマンド，ステートメント）を入力したときは「プログラムの記憶のさ

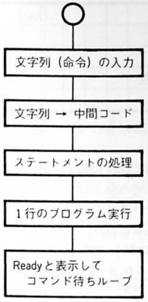


図2·14　ダイレクトモード時のフロー

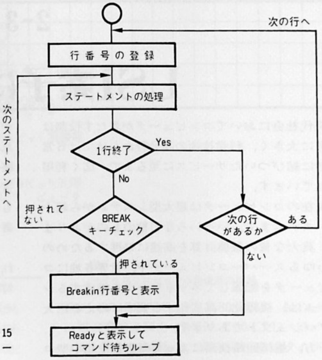


図2·15
プログラムモード時のフロー

れ方」と同様の処理から始まります（図2·14）．

文字列と中間コードはBASICテキストエリア（BASICのプログラムのエリア）とは別の場所に格納して実行します．

b．プログラムの実行

ダイレクトモードで RUN CR としたり，GOTO“行番号” を実行すると，プログラムの先頭，またはプログラムの途中から実行を開始します(図2·15参照)．

ステートメントの処理でEND文を実行すると処理は終了し，コマンド待ちループに入ります．

c．ステートメントの処理

ステートメントの処理は，中間コードを元に各サブルーチンへジャンプすることで各命令のサブルーチンが処理します．

変数名が書いてあり，その後に代入文や式が書いてあれば，変数名や式の計算，代入文の処理をします．その中で関数の処理があれば中間コードを元に，その処理へジャンプします．

GOTO, IF〜THEN, PRINT などの処理も同様に，中間コードを元に各処理のサブルーチンへジャンプします．

各サブルーチンへジャンプするときは，サブルーチンからの戻り先などを記憶しておく必要があります．また，FOR〜NEXT で繰り返しの処理を行うための値（行番号，増分，終了値など）をまとめて記憶する必要もあります．このような一時的な値を記憶しておく場所をスタック(stack)といいます．

各ステートメントのサブルーチンは，これらスタックをうまく使い，プログラムをスピーディに実行できるようになっています．

d．ジャンプテーブルとサブルーチン

「ステートメントの処理」で，BASICインタープリタは中間コードをもとにジャンプすることを述べましたが，これには，ジャンプテーブルという手法を用いています．

BASICの中間コード（表参照）は，1バイト，FE＋1バイト，FF＋1バイトの3種類があります．ジャンプテーブルとはメモリ上に記憶された表のようなもので，中間コードのB5Hは××××番地，97Hは△△△△番地といった中間コードに対応するアドレス（番地）をすぐに参照することができます．

これがもし中間コードではなくて，命令語をアスキーコードで文字として記憶していればどうなるでしょうか．命令語の文字長は不統一なので，命令語に対応するサブルーチンのアドレスを知るには，ひとつひとつ表の中身とチェックしなければなりません．文字長の長い命令語（PRINTやRETURNなど）のせいでメモリもたくさん必要になります．これに対し，最近のBASICはすべて中間コードを採用しているので，処理の高速化とメモリの節約が実現されています．

2-3 LSI素子について

現代社会においてコンピュータが果たす役割は非常に大きく，科学技術の諸分野からはては日常生活に結びついたサービスに至るまで，広く利用されています．

現在のコンピュータは超大型システムから超小型システムまで，いろいろな形態のものがあります．莫大な量の数値計算を高速に処理するためのいわゆるスーパーコンピュータや，世界各地にコンピュータを配置してネットワークを構成するシステムは，複雑化，高度化する要求に応える巨大なプロジェクトであります．

一方，集積回路技術によって誕生した超小型コンピュータ，すなわちマイクロコンピュータは，そのミクロな形態と，大量に安価に生産できる高い経済性を有し，機器組込み制御部品としての単能的な利用から，初期のコンピュータの機能をはるかにしのぐような汎用システムまで広く，かつ急速に現代社会に浸透しています．

マイクロコンピュータのこれまでの飛躍的な発展の背景には，その形態，すなわちLSIの技術革新があります．集積技術，高速化，低消費電力化が，機能の充実や低価格化と密接なかかわりを持っているのです．

特に重要な集積度の向上は，図2･16に示すMoorの曲線(Moor's curve)によって顕著な推移を見ることができます．図からもわかるようにこの曲線はICからLSI, さらにVLSIへと推移するチップの集積度の向上を表わしたもので，1960年当時1チップ＝1素子のデバイス技術であったものが，ほぼ1年ごとに2倍になっていた模様が示されています．その要因としてはデバイス構造，回路構成の工夫や微細加工技術の相互の発展による寄与分と，チップサイズの向上によって結果的にもたらされる寄与分とが考えられ後者が約1／4寄与しているといえるでしょう．

マイクロプロセッサに関しては，図2･17に見られる集積度の向上がみられます．Moorの曲線より勾配はゆるやかです．マイクロプロセッサでは，その機能，半導体技術によってかなり明確な世代の交代がみられ，同時にそれが合わせて示されてます．第1世代は1971年にインテル社が発表したPMOSの4ビットCPU 4004を代表とするもので，電卓にその起源を持っています．これは機能的には第2，第3世代へと移る過程で4ビットの1チップマイクロコンピュータとして受け継がれていきました．第2世代では，Nチャネル MOSが主体となり，4ビット1チップマイクロコンピュータのほかに，汎用8ビットCPU，インテル社の8080，モトローラ社のMC 6800などが生まれました．第3世代はザイログ社のZ 80から入ったと考えられます．このZ 80は8ビットCPUの1つの工業基準を作ったと考えられています．4ビット1チップマイクロコンピュータは，この世代に入って低消費電力のCMOSタイプが主流となってきました．第2，第3世代ですでに12ビット，16ビットCPUが出ていますが，1978年～1979年にかけてインテル社の8086，ザイログ社のZ 8000，モトローラ社のMC 68000などの高機能マイクロプロセッサが発

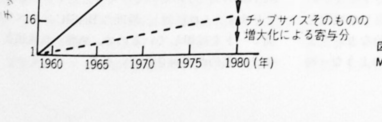


図2･16
Moorの曲線

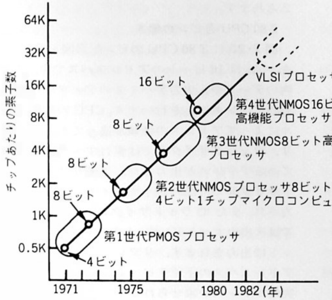


図2・17
マイクロプロセッサの集積度の推移

表され，1980年代に入って大いに活用されてくるにいたっています．そしてこれを第4世代と見ることができます．LSIの集積度は，現在なお同スピードで向上しており，VLSI時代へ向けてさらに高機能のプロセッサが出現するとみられます．その基本は語長の拡大，命令機能の充実，大容量記憶の管理，高速処理，高度な入出力コントロールなどです．

それでは，具体的にMZ-1500で使われているLSIなどの説明を致しましょう．

2-3-1 Z80CPU

Z80CPUはZ80ファミリーの中心となるものであり，高度な処理能力を備えた第3世代のマイクロプロセッサです．

8ビット並列処理のワンチップ・マイクロプロセッサであるZ80CPUには12の汎用レジスタB, B′, C, C′, D, D′, E, E′, H, H′, L, L′があります．この汎用レジスタはBC, DE, HLというように2つを組合わせて16ビットレジスタとして使用することもできます．IX, IYというインデックスレジスタが2つあり，他には割込みベクトルI，メモリ・リフレッシュレジスタR，スタックポインタSP，プログラムカウンタPC，フラグレジスタF，そしてアキュムレータAがあります（図2・18参照）．

Z80CPUの外部端子にはアドレスバス16本，データバス8本，システム制御線5本，バス制御線2本，それにクロックと電源が出ています．

クロックは単相でよく，Z80CPU　Aバージョンは周波数の上限を4 MHz, Z80CPU　Bバージョンは上限を6 MHzまで保証した高速タイプです．

Z80CPUの命令体系は図2・19に示されている通り，Intel 8080Aの命令をすべて含み，さらに8080Aにはない多くの新しい命令やアドレッシングが加えられています．

このZ80CPUの特徴的なものを列記しますと以下のようになります．

① 基本命令が158種と多い（8080Aの78種の全命令を含み，機械語において互換性がある）
② 内蔵レジスタが多く，22のレジスタを持っている
③ 強力な割込み機能を持っている
　○マスク可能割込み＝3モード
　○マスク不能割込み
④ 標準的なスタティックメモリやダイナミックメモリと直結するのに外部回路をほとんど必要

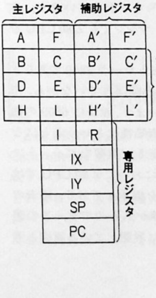


図2・18
Z80レジスタの構成

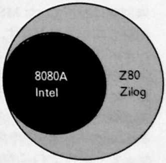


図2・19　Z80命令体系

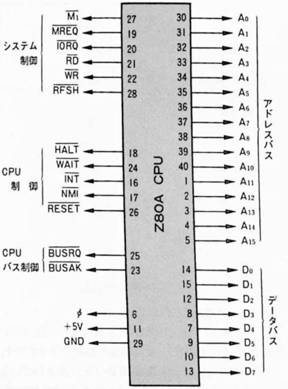


図2・20　Z80のピン配置

としない．リフレッシュ回路を内蔵している(後述参照)

⑤ 命令フェッチ速度はそれぞれ1.6μs (Z80)，1.0μs (Z80A)，0.67μs (Z80B) となっている．

⑥ ＋5Vの単一電源及び単相クロックだけを必要とする

⑦ 入出力端子はすべてTTLコンパチブル

⑧ パッケージは40ピンDIP(dual inline package)となっている

市場での現状をみますと，現在日本で使用されているCPUの中でZ80CPUはとりわけ占有率が高く，たくさんのユーザーをかかえています．先頃話題になっていたMSXマシンでもZ80CPUを採用しているほどです．

次に占有率が高いのはモトローラ6809であります．究極の8ビットCPUと呼ばれていながら，市場では第2位に甘んじています．確かに6809の機能は高いけれども，いざ市場性はということになると，ソフトウェアの豊富さが重要なポイントになってくるのです．この点においてZ80CPUでは8080以来のソフトウェアを継承しており，過去の膨大なソフトウェアの蓄積があるのです．この差が市場でZ80の方が有利に展開している原因と考えられます．

Z80CPU各ピンの働き

図2・20にZ80CPUのピン配置図を示します．A_0～A_{15}は16ビットのアドレスバスです．メモリ内のデータや入出力デバイスのデータの送受のためのアドレス指定を行います．CPUの動作サイクルによってアドレスの種類が違ってきます．つまり，フェッチサイクルでは実行すべき命令の入っているアドレスが出力され，メモリリード，ライトサイクルでは読み書きするメモリアドレスが出力され，またIOライトサイクル，リードサイクルでは入出力すべきIOポートのアドレスが下位8ビットに出力されます．リフレッシュサイクルではアドレスバスの下位7ビットにリフレッシュ用の実効アドレスが乗せられます．

D_0～D_7は8ビットの**双方向性データバス**です．データバスはメモリ，入出力間，あるいはメモリと入出力間でデータ交換を行うために用います．このデータバスもフェッチサイクルではメモリから命令コードが乗せられ，メモリリードサイクルではメモリからデータが乗ってきます．メモリライトサイクルとIOライトサイクルではCPU内のレジスタからのデータがデータバス上に乗せられます．

このようにアドレスバス，データバスとも種々別々の目的をもった信号が次々に乗せられるので，今の信号が何を意味するかを識別するためにシステム制御線があります．$\overline{RD}$, $\overline{MREQ}$, $\overline{IORQ}$, $\overline{M\ 1}$, $\overline{RFSH}$という6つのシステム制御線の組合わせによって識別されます．

その他にCPUには単独で意味する信号が7つあります．

$\overline{WAIT}$	クロックサイクル数を増やす
$\overline{INT}$	割込み要求
$\overline{NMI}$	マスク不能割込み要求
$\overline{RESET}$	初期状態に戻す
$\overline{BUSRQ}$	CPUの動作を停止させバスを空渡す要求
$\overline{BUSAK}$	$\overline{BUSRQ}$を受付けた返事
$\overline{HALT}$	CPUがホルト命令を実行し，停止状態に入ったことを外部へ知らせる

☆Z80のメモリ空間と IO空間

Z80では通常接続できるメモリは最大64KB(バイト)までです．他の8ビットCPUも大体同じです．これはつまりZ80CPUにはアドレスバスがA_0

$\sim A_{15}$まで16本ありますので，アクセスできるメモリ領域は$2^{16}=64$ Kとなるからです．このようなメモリとは別にIOの領域というものがあります．IOの領域として割付けられるのは，0～255までの256個のアドレスまでです．つまり入出力のサイクルではアドレスバスの下位8ビット($2^8=256$)のみ使用しているためです．

メモリ空間も，IO空間も同一のアドレスバスを使用しますが，$\overline{MREQ}$信号と$\overline{IORQ}$信号で切換えることにより，メモリ空間とIO空間というものが別々に存在するというわけです．

メモリ空間とのアクセスはLD命令，IO空間とのアクセスはIN命令，OUT命令を使います．

☆機能について

Z80 CPUには次のような機能があります．

8ビットロード
16ビットロード
レジスタ交換
メモリブロック転送
メモリブロックサーチ
8ビット演算
16ビット演算
ローテイト，シフト
10進補正
フラグ操作
CPU制御
ビット操作
ジャンプ
コール，リターン，リスタート
入出力，連続入出力
オートリフレッシュ機能

Z80 CPUはメモリ・リフレッシュカウンタを内蔵していますので，スタティックメモリと同じ手軽さでダイナミックメモリを使用することができます．

リフレッシュサイクルは各フェッチサイクルの後半で行なわれます．アドレスバスの下位7ビットにはRレジスタの内容が出力されます．このRレジスタは各命令のフェッチごとに自動的にインクリメントされます．

このときは$\overline{RD}$，$\overline{WR}$の信号の代わりに$\overline{RFSH}$信号が出て，リフレッシュサイクルであることを示します．

☆割込み機能について

ここでは割込みの概念的なことだけ触れることにして，具体的には次章で触れることにします．

CPUが通常のプログラムを実行している途中に，急に周辺デバイス等から他の処理が必要になったときに，$\overline{INT}$信号や$\overline{NMI}$信号を入れることによって，通常のプログラムとは関係なく特定のアドレスにある周辺デバイス処理ルーチンへジャンプさせることができます．これを割込みと呼びます．

CPUの処理速度に比べて，プリンタ等の周辺デバイスの動作速度ははるかに遅いものです．このとき単にCPUを待たせておくのではなく，CPUに別の仕事をさせることもできます．

このように割込みは，ソフト的な要求で呼びだされるサブルーチンコールとは区別され，ハード的な要求によるものです．普通この処理ルーチンにはCPUと周辺とのデータ，ステータス，あるいは制御情報の送受などが含まれています．処理ルーチンが終わると割込みがかかる前と同じ状態に戻ります．

☆スタックについて

メモリには普通の意味での記憶領域の外にスタックという特別な領域を設定することができます．

スタックはLIFO(last-in-fast-out)形式のファイルとして構成されます．LIFOとは最後にPUSHしたデータが，取り出す時には最初にPOPが行なわれるということです．つまり，スタックに積上げられたデータは必ず一番上からしか取り出せません．スタックポインタは外部RAM上のスタック領域のその時点での最上位のアドレスを保持します．

スタックは，多重レベルの割込みや，無限のサブルーチン・ネスティングを簡単にするのに有効です．

2-3-2 Z80PIO

Z80 PIO(Parallel Input／Output Interface Controller)はZ80 CPUと周辺機器(キーボード，紙テープリーダ，紙テープパンチャ，プリンタ等)との8ビット並列入出力インターフェイスコントローラであり，プログラム可能なTTLコンパチブルの入出力ポートを2個持っています．

Z 80 PIO を CPU のインターフェイスデバイスとして用いることで，余分な外部回路を追加しないで広範囲の周辺装置に適用することができます．

PIO の動作には各ポートごとに次の 4 つのモードがあります．

モード 0　出力モード
　　　1　入力モード
　　　2　入出力モード（A ポートのみ）
　　　3　ビットモード

モード 0 ～ 2 はハンドシェイクにより 8 ビットデータを入出力します．モード 3 ではビット単位に入出力を選択できます．

Z 80 PIO のおもな特徴は

① データ転送の制御用のハンドシェイク線をもつ 8 ビット双方向性の周辺装置用インターフェイス・ポートがあります
② 周辺機器の状態によるプログラム可能な割込み機能があります
③ 各ポートごとに 4 つの動作モードがありますこれらにはすべてハンドシェイクで制御される割込み機能があります
④ 自動的に割込みベクトルを出せるデイジーチェイン構造の割込み優先回路を内蔵しています
⑤ ポート B の出力はダーリントン・トランジスタを駆動できます
⑥ 全入出力は TTL コンパチブルです

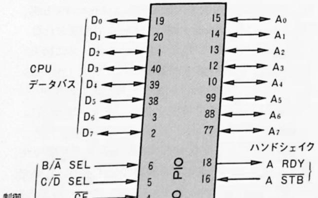


図 2・21　Z 80 PIO のピン配置図

項　目	記号	定　格　値	単位
入力電圧	Vin	−0.3 ～ +7.0	V
出力電圧	Vout	−0.3 ～ +7.0	V
動作温度	Topr	0 ～ +70	℃
保存温度	Tstg	−65 ～ +150	℃

表 2・7　Z 80 PIO の最大定格

⑦ 5 V 単一電源，単相クロックのみで動作します

Z 80 PIO は他のインターフェイスとは異なり，周辺装置と CPU との間のデータ転送を割込み制御の方法により効率よく行なうことができます．また Z 80 PIO には Z 80 CPU の効率のよい割込み処理能力を入出力データの転送期間に充分に発揮できるように，割込み処理用の，論理回路が組込まれており，かつ多重の割込みをネストさせるために必要な論理回路も内蔵しています．このため割込み処理用の余分な外部回路を必要としません．

その他，周辺装置が特定の状態になったときに，割込み信号が出るようにプログラムしておくこともできるようになっています．

この割込みの能力によってプロセッサが周辺装置のステータスを調べる間の時間を節約するのに役立ちます．

Z 80 PIO の定格を表 2・7 に示します．また，Z 80 PIO のピン配置図を図 2・21 に示します．

2-3-3 Z80PIOのアーキテクチャ

図 2・22 に Z-80 PIO のブロック図を示しますが次の各部で構成されています．

- Z-80 A CPU 用バスインターフェイス
- 内部制御回路
- ポート A 入出力論理回路
- ポート B 入出力論理回路
- 割込み制御回路

CPU バスインターフェイス用論理回路は，他の外部回路を追加せずに，直接 CPU に接続できるようになっていますが，大きなシステムでは必要に応じてアドレスデコーダやラインバッファを付加しなければならないこともあります．

内部制御回路は CPU データバスを周辺装置用インターフェイス部（ポート A，B）と同期させるた

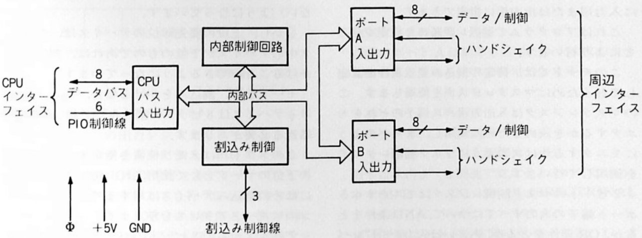


図2・22　Z80 PIOのブロック図

めのものです．２つの入出力ポート（A, B）は，ほぼ同様に使用でき，直接周辺装置と結合する部分です．

ポート入出力論理回路は図2・23に示すようにハンドシェイク用論理回路を含む６つのレジスタから構成されています．これらはそれぞれ８ビットデータ入力レジスタ，８ビットデータ出力レジスタ，２ビットのモード制御用レジスタ，８ビットのマスクレジスタ，８ビット入出力選択レジスタ，２ビットのマスク制御用レジスタです．

２ビットのモード制御用レジスタは４つの動作モード（バイト入力，バイト出力，バイト双方向バス，ビット制御モード）のうちいずれかを指定するためのもので，CPUからの制御で設定されます．

周辺装置とCPUとの間のデータ転送はすべて入出力用レジスタを介して行われ，CPUからの出力データは出力レジスタを通して周辺装置へ送られ，CPUへの入力データは入力レジスタを通してCPUに読込まれます．データの入出力はどの時点で行ってもかまいません．

各ポートのハンドシェイク線はPIOと周辺装置との間のデータ転送の制御用として用います．

８ビット・マスクレジスタと８ビット入出力選択用レジスタはビット制御モードのときにのみ使用されます．このモードでは，周辺と結ばれるデ

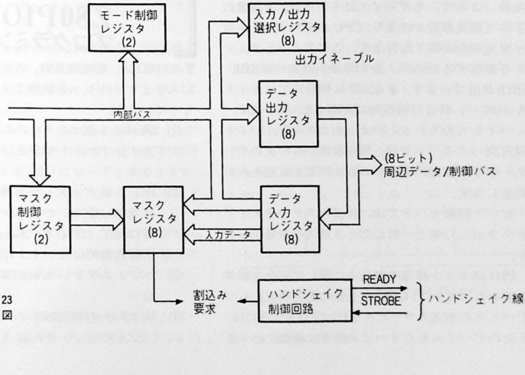


図2・23
ポート入出力ブロック図

ータ転送，もしくは制御用の8本のバス端子を個々に入力用または出力用に指定できます．

これはプログラムで選択レジスタを設定することにより行います．

このモードでは，特定の割込み発生条件を設定しておくためにマスクレジスタを使用します．このマスクレジスタは入出力用バス端子のどれをモニタするかを決めるために使用し，またどのようにモニタするかは次のようにマスク制御レジスタを使用して行います．

2ビットのマスク制御レジスタはモニタすべきポート端子の入力すべてについて，AND条件をとるか，OR条件をとるか，あるいは入力が“H”レベルのときアクティブとするか，“L”レベルのときアクティブとするかを規定するためのものです．これらの指定に合ったステータスになれば，割込みが発生します．

このように，あらかじめ条件設定をプログラミングしておく方式では，周辺装置の特定のステータスで自動的に割込みを発生させるので，CPUが周辺のステータス・チェックをする必要がないという利点があります．システム内に3種の緊急条件がある場合，プログラムでこのうちの1種でも起これば（OR条件），あるいは3種すべてが起これば（AND条件）割込みを発生させる，といった使い分けもできます．

割込み制御用論理回路はCPUの割込みに関するすべての処理を行い，優先順位の決定などについても関与している部分です．デイジーチェインの接続によって，各デバイスはその物理的な位置によって優先順位が決まり，CPUに近いデバイスほど優先順位が高くなります．このデイジーチェインを形成するために，各PIOから2本の線(IEI, IEO)が出ています．また，同一PIO内ではポートAがポートBより優先順位が高くなっています．

バイト入力モード，バイト出力モード，バイト双方向バスモードでは，周辺装置からの次のデータ・バイト転送の要求のあるたびごとに割込みが発生します．

ビット制御モードでは，周辺のステータスがプログラムした値と一致したとき割込みが発生します．

PIOはネスト構造の割込みに関して完全な制御機能をもっていて，CPUが上位の優先順位をもつデバイスの割込みサービスを行っているときには，下位のデバイスからサービス要求は通さない（下位の優先順位のデバイスから，割込みを発生させない）ようになっています．

しかし，上位の優先順位のデバイスは，サービス中のデバイスが下位のものであれば，割込みをかけることができるようになっています．

モード2で，割込みをかける場合，割込みをかけるデバイスは8ビットの割込みベクトルをCPUに送る必要があります．

このベクトルはメモリ位置を指定するポインタの下位のデータとして使用され，指定された位置にはその割込みデバイスに対するサービスルーチンのスタートアドレスを与えます．

デバイスからの8ビットのベクトルはポインタ（間接）の下位8ビットとなり，CPUのIレジスタのデータが，そのポインタの上位8ビットとなります．

なお，PIOの各ポート（AとB）の割込みのベクトルはプログラムによって指定するものである点に注意してください．

ベクトルの最下位ビットはスタートアドレスを連続する2バイト（16ビット）で指定しなければならないので，プログラマ側で0を指定しなければなりません．

PIO自身でCPUデータバスに乗るRETI（割込みからのリターン）命令を直接解読できるので，システム内の各PIOはCPUとの間に別の通信線を引かなくても，CPUの割込みサービスが終了する時期を検知できるようになっています．

2-3-4 Z80PIOのプログラミング

Z-80 PIOは，電源投入時，自動的にリセット状態に入ります．リセット状態では次のような機能をもっています．

① 両ポートのマスクレジスタがリセットされ，全データビットが禁止（インヒビット）になる

② ポートのデータ・バス線は高インピーダンス状態になり，ハンドシェイクのレディ信号は非アクティブ（“L”）となる．動作モードは自動的にモード1に設定される

③ ベクトルアドレスレジスタはリセットされない

④ 両ポートの割込みイネーブルフリップフロップがリセットされる

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	V_2	V_1	0

この制御語により割込みベクトルを指定する（D_0 = 0）

図2・24　割込みベクトルのフォーマット

⑤　両ポートの出力レジスタもリセットされる

電源投入時のリセットに加え，PIOのリセットは，RDあるいはIORQ信号を非アクティブにして2クロックのM1を与えても行うことができます．

M1期間にRDかIORQかが検出されなければ，PIOはM1信号が非アクティブになった直後リセット状態になります．

このリセットの目的は電源を切ってリセットするという手順を省き，簡単な外部ゲートのみでリセットを行うことができるようにするためであり，40ピンのパッケージに収める上での処置です．

一度PIOが内部リセット状態に入ると，CPUから制御語を受けるまでそのままリセット状態を保持します．

割込みベクトルのローディング

PIOは，モード2の，割込み応答のとき，Z-80 CPU側に情報を返して，CPUがこれを処理・操作できるように設計されています．

このモードでは，割込み要求しているデバイスから割込みベクトルを供給することになりますが，このベクトルはCPU内でそのデバイスの割込みサービスルーチンのアドレスを形成するのに使用されます．

このベクトルは，その時点で最高位の優先順位をもっているデバイス（PIOなど）から割込みアクノリッジ・サイクル期間中に，Z80 データバス上に乗せられます．

PIOが出す割込みベクトルは図2・24のようなフォーマットで，CPUから出力命令によって制御語を書き出し，希望するPIOにロードしておきます．この場合のポート・アドレスはAまたはBポートの制御語に対応するアドレスです．

ベクトルの書込みにおいては，D_0はフラグビットとして使用され，このビットが0である場合のV_1～V_7までが，PIO内のベクトルレジスタにロードされます．

割込みアクノリッジ期間に割込みをかけたポートのベクトルが，Z-80 データバス上に図2・24のようなフォーマットで現われます

動作モードの選択

PIOのポートAは前にも述べたように次のような4種の異なったモードで使用できます．

モード0　（出力モード）
モード1　（入力モード）
モード2　（双方向モード）
モード3　（ビット制御モード）

数字は次のような関連で覚えるとよいでしょう．

0 = OUT,　1 = IN,　2 = BI-DIRECTIONAL
（出力）　（入力）　（双方向）

ポートBはモード2を除いたモード0，1，3で動作可能です．動作モードの指定はPIOに図2・25のようなフォーマットの制御語を書込んで行います．

ビットD_7, D_6により，以下のようにモードが指定できます．

D_7	D_6	モード
0	0	0（出力）
0	1	1（入力）
1	0	2（双方向）
1	1	3（制御）

ビットD_5, D_4は無視されます（0，1いずれでもよい）．

ビットD_3～D_0により動作モードを指定し，1111に設定しておかなければいけません．

モード0を指定した場合，ポートの出力レジスタ内のデータが対応するポートのデータバス上に乗せられます．この出力レジスタの内容は，出力命令を用いて新しいデータについても簡単に書換えられますし，また，ある時点での出力レジスタの内容を入力命令を用いてCPU側へ呼び戻す（出力レジスタの内容は変化しない）こともできます．

モード0では，CPUから出力レジスタに対してデータの書込みがあった場合，そのポートのハンドシェイク線であるレディ信号はアクティブ（"H"）となり，周辺装置にデータの転送が可能であると

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
M1	M0	X	X	1	1	1	1

モード語（D_7, D_6）

このデータフォーマットのときモード指定が行われる（D_3～D_0）

X=使用しないビット
（0,1いずれでもよい）

図2・25　動作モードの指定

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
I/O_7	I/O_6	I/O_5	I/O_4	I/O_3	I/O_2	I/O_1	I/O_0

図 2・26
入出力の指定

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
割込みイネーブル	AND / OR	High / Low	マスク指定	0	1	1	1
モード3のみで使用				割込み制御語を指定する			

図 2・27
ポートへの割込みの設定

いう指示を出します．この信号は周辺装置から受信したというストローブ信号が返されるまで，“H”レベルのままです．その返されたストローブの立上がりエッジで，もし割込みが発生するようにプログラムされていると，割込みが発生し，レディ線は非アクティブ(“L”)となります．この簡単なハンドシェイク方式は多くの周辺装置によく使用されているものと同じ方式です．

モード1を指定した場合，ポートは入力モードとなります．ハンドシェイク動作を始めるためには，まずそのポートに対して，読出しを行うだけでよいのです(ダミーリード)．この動作によってレディ線がアクティブとなり，“周辺装置へ入力レジスタが空になったのでデータをロードせよ”という指示が出されます．装置はストローブ信号に同期しデータを送出しますがPIOはこのストローブ信号がアクティブになることによって内部入力レジスタにデータを取込みます．この返されたストローブの立上がりエッジで，可能であれば割込みが発生し，レディ信号が非アクティブ状態に落されます（“L”となる)．データがオーバラン状態にならないように注意が行き届いていれば，レディ信号の状態にかかわらず，ストローブをかけてデータを入力レジスタへロードしてもかまいません．

モード 2の双方向データ転送はすべてのハンドシェイク線(A，B各2本)を使用して行います．したがって，モード2では，ポートAだけをデータの入出力用として使用し，ハンドシェイク線はポートAの2本を出力制御用とし，ポートBの2本を入力制御用として使用します．A RDY，B RDYは両方とも，同時にアクティブとなります．モード0とモード2における出力時の動作の違いは，このモード2では双方向にデータ転送を行うのを可能にするためA STBがアクティブになったときにだけポートAの出力レジスタ上のデータがポートAのデータバス上に乗せられる，という点だけです．

モード3の動作はステータスを調べるためと制御用として応用するためのものであり，この場合ハンドシェイク信号は使用しません．モード3を選択すると次にポートアドレスに書込む制御語によって，ポートのデータバス線のどれを入力用とし，どれを出力用とするかを指定する必要があります．あるビットを1に設定すれば，そのビットに対応するデータバス端子は入力用となり，0に設定したビットに対応するデータバス端子は出力用となります（図 2・26 参照)．

モード3の動作時，ストローブ信号は無視され，レディ線は“L”レベルに保持されます．ポートとCPU間のデータのやりとりは動作中任意の時点で行えます．ポートを読む場合，CPUに入るデータは入力として指定されたポートのデータバス線からのデータと，出力として指定されたポートのデータバスからのデータが重なり合っています．

割込み制御語の設定

各ポートの割込み制御語の設定は図 2・27 のようなフォーマットで行います．

ビット $D_7=1$ のとき，対応するポートの割込みイネーブル・フリップフロップがセットされ，ポートからの割込み要求が可能となります．$D_7=0$ のとき，イネーブル・フラグはリセットされ，割

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
MB_7	MB_6	MB_5	MB_4	MB_3	MB_2	MB_1	MB_0

図 2・28
マスクの設定

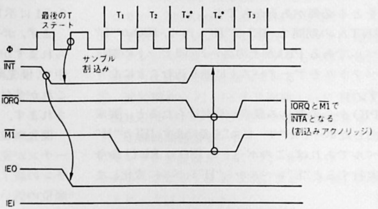


図 2・29
割込みアクノリッジのタイミング

込みを発生できません．イネーブル・フラグがセットされていて，割込みが保留状態ならば，CPUの割込み要求信号であるINT線は"L"レベルのままです．ビット D_6, D_5, D_4はモード3でのみ使用されます．割込み制御語のビット D_4がセットされると，どのモードで動作している場合でも，未処理の割込みはすべてリセットされます．モード3で入出力線群のどれかがある指定状態になったとき，割込み動作を実行できるようにするための条件設定にこの3ビットを用います．ビット D_6でモード3のデータ一致条件をANDで採るか，ORで採るかを定め，$D_6 = 1$ならばAND，$D_6 = 0$ならばORとなります．たとえば，ANDの場合，入力線の指定ビットがすべてアクティブのときにのみ，割込み発生の条件となり，ORの場合，指定ビットのいずれかがアクティブならば，割込み発生の条件となります．

ビット D_5は入力線のアクティブ状態の極性規定をするためのもので $D_5 = 1$ならば入力線が"H"レベルのときアクティブとみなし，$D_5 = 0$ならば，入力線が"L"レベルのときアクティブとみなします．

ビット $D_4 = 1$のとき，同一ポートに図2・28のように書込まれた制御語はマスクと解釈されます．

このマスクでビット0の箇所の入力線のみを調べ，設定条件と一致すれば割込みを発生させる条件となります．

割込みサービス

PIOから割込み要求を受け取ったあと，CPUは割込みアクノリッジを送出します．(このアクノリッジ信号としてM1とIORQが同時に出る．)

この期間において，PIO内の割込み論理回路は割込み要求を出している最も優先順位の高いポートを決定します．(これは単にIEIが"H"レベルでIEOが"L"レベルのデバイスを求めることで決められる．)

デイジーチェインの，イネーブル線の状態を確実にするため，M1がアクティブになったとき，各デバイスの割込み要求状態の変更は許されません．

割込みアクノリッジの期間に，割込みを出している最も優先順位の高いデバイスがCPUデータバス上にその割込みベクトルを乗せます．

図2・29に割込み要求に関係したタイミングを示します．M1期間では，新しい割込み要求を発生させることができません．これは4個までのPIO

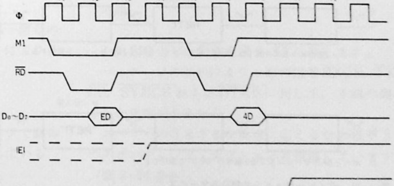


図 2・30
割込みサイクルからの復帰

からの割込みイネーブル信号を受けるに十分な時間をとる必要があるからです．

INTA の期間で，IEI が“H”レベル，IEO が“L”レベルである PIO がそのポートの 8 ビットの割込みベクトルをデータバス上に送り込むことになります．

PIO からの割込み要求が受付けられると，要求を出したポートが“サービス”を受けます．IEI が“H”レベルであれば，このポートの IEO は RETI 命令を実行すると“L”レベルから“H”レベルに変化します．

割込み要求が受付けられていない PIO は OP コード“ED”（入出力命令などの OP コード）を解読後，次の M 1 の 1 サイクル間 IEO を強制的に“H”レベルにします．

この動作により，この 2 バイトの RETI 命令を選ばれた PIO ポートだけが解読することができるようにしています（図 2･30 参照）．

4 つのポートをデイジーチェインで接続してある場合のネスト構造の割込みシーケンスの例を図 2･31 に示します．

まず，ポート 2 A が要求を出し割込みが受付けられますが，このポートがサービスされている途中で，優先順位の高いポート 1 B が要求を出すと，これが受付けられ，ポート 2 A のサービスは保留されます．

優先順位の高いポート 1 B に対するサービスルーチンが完了し，RETI 命令を実行して，そのルーチンの終了をポート 2 A に知らせます．次に優先順位の低いポート 2 A のサービスルーチンを再開します．

2-3-5 Z80SIO

Z 80 SIO(Serial Input／Output Controller)は，2 チャネルの多機能周辺デバイスであり，マイクロコンピュータシステムで必要とされる，広範囲

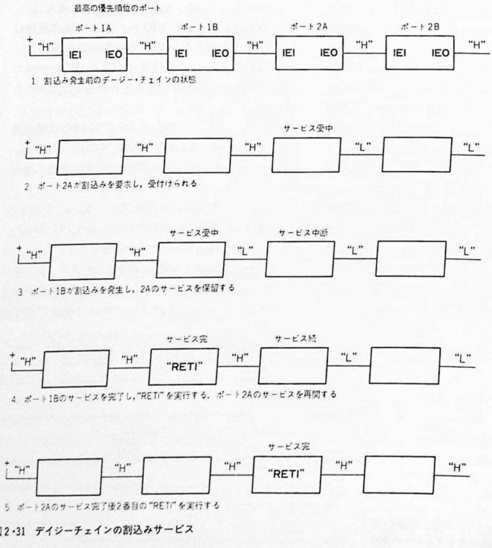


図 2･31 デイジーチェインの割込みサービス

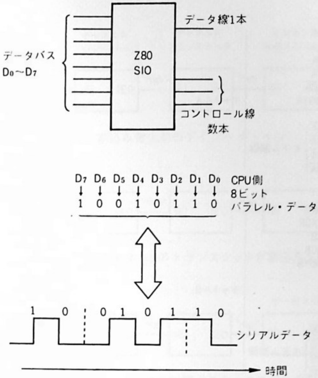


図2・32　Z80 SIOの基本的動作

のシリアルデータ通信の要求を満足するように設計されています．例えば，電話回線を使って長距離の通信を行うときなどに必要とされます．SIOの主な機能は，直列ー並列，並列ー直列変換ですが，種々のシリアルデータ通信への応用に対してもソフトウェア制御によって効果的に使用できます．

この SIO は図2・32のように，データを直列，すなわち時間経過に従って，1本のデータ線で入出力するポートコントローラです．

直列データ伝送の方式には，同期・非同期・バイト指向・ビット指向などがあり，Z80 SIOはこれらを処理できます．

SIOはどのような同期モードに対しても，CRCを生成および照合する能力をもっています．また，2つのチャネルの両方ともモデム制御線をもっていますが，これらの制御線を必要としない場合には，汎用の入出力線として使用できます．

Z80 SIOは，その動作周波数および端子信号の取出し方法（ボンディングオプション）により3種類用意されています．

Z80 SIOのおもな特徴は

① パッケージは40ピン DIP

② ＋5 Vの単一電源および単相クロックで動作

③ 独立した全二重チャネルを2個もっている

④ データ転送速度は次の3種がある

○ 0〜500 Kbps（2.5 MHzのとき）

○ 0〜800 Kbps（4.0 MHzのとき）

○ 0〜1,200 Kbps（6.0 MHzのとき）

⑤ 受信データレジスタは4段バッファ，送信データレジスタは2段バッファをもっている．両方とも LIFO形式となっている

⑥ 非同期モード，バイナリ同期モード，HDLC／SDLCモードの3つのモードを使用できる

(1)非同期通信モード

○キャラクタ長は5，6，7，8ビット

○ストップビットは1，1½，2ビット長，スタートビットは1ビット

○パリティビットは偶数，奇数，パリティなしのどれでも指定できる

○クロックモードは ×1，×16，×32，×64がある

○ブレークの発生と検出

○パリティエラー，オーバーランエラー，フレーミングエラーの検出

(2)バイナリ同期通信モード

○内部または外部キャラクタ同期が可能

○モノシンク，バイシンク通信が可能

○同期キャラクタの自動挿入および削除

○ CRCを生成および照合

(3) HDLC／IBM-SDLCモード

○アボートシーケンスの生成と検出

○ゼロの自動挿入および削除

○フラグの自動挿入

○アドレスフィールドの検出

○1ビットから8ビットのキャラクタが使用可能

○受信メッセージのオーバーランからの保護

○ CRCの生成および照合

(4)割込み

○デイジーチェイン割込み制御回路の採用による割込み時のベクトル送出自動処理をする

○割込みベクトルのプログラムが可能

○高速割込み処理のために，ステータス情報により割込みベクトルを自動的に修飾（一部変更）可能

Z80 SIOのピン配置を図2・33に示します．

パッケージの制約により4個の信号$\overline{RXCB}$，$\overline{TXCB}$，$\overline{SYNCB}$および$\overline{DTRB}$に対して，3個の端子しか使用できません．

ユーザーのさまざまな要求に応えるため以下3つのボンディング・オプションを提供しています（図2・34参照）．

Z80 SIO／0

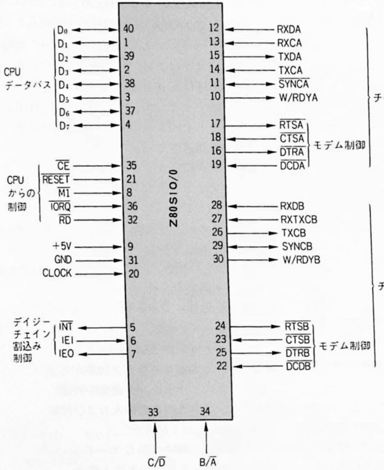


図2・33
Z80 SIOのピン配置図

4個のすべての信号をもっていますが, $\overline{\text{RXCB}}$と$\overline{\text{TXCB}}$を同じ端子にボンディングし $\overline{\text{RXTXCB}}$としています

Z80 SIO／1

$\overline{\text{DTRB}}$を除去し $\overline{\text{RXCB}}$と $\overline{\text{TXCB}}$を別々の端子に結線し, クロック倍率の異なる場合の要求に対処しています

Z80 SIO／2

$\overline{\text{SYNCB}}$を除去し, $\overline{\text{RXCB}}$, $\overline{\text{TXCB}}$, $\overline{\text{DTRB}}$の端子をもっています

Z80 SIOの定格を表2・8に示します.

項　目	記号	定　格　値	単位
入力電圧	Vin	−0.3～+7.0	V
出力電圧	Vout	−0.3～+7.0	V
動作温度	Topr	0～+70	℃
保存温度	Tstg	−65～+150	℃

表2・8　Z80 SIOの最大定格

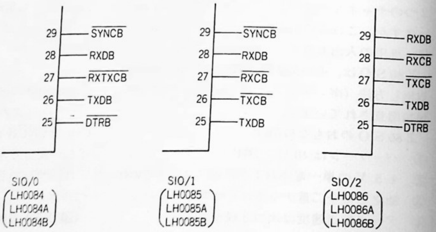


図2・34
ボンディングオプション

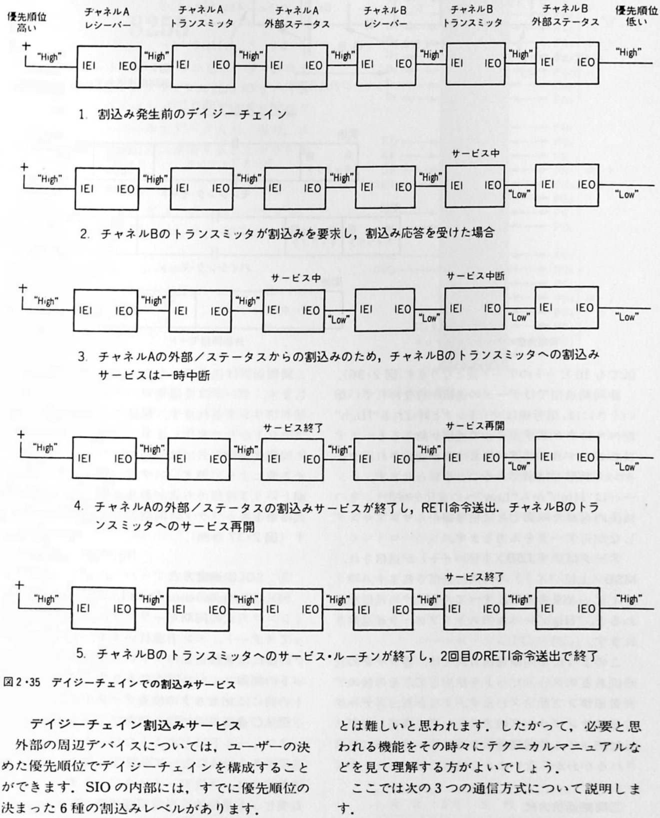


図2・35　デイジーチェインでの割込みサービス

デイジーチェイン割込みサービス

外部の周辺デバイスについては，ユーザーの決めた優先順位でデイジーチェインを構成することができます．SIOの内部には，すでに優先順位の決まった6種の割込みレベルがあります．

図2・35は，チャネルBのトランスミッタが割込みサービスを受けているときに，チャネルA(優先順位が高い）から割込みがかかった場合を示しています．

Z80SIOは非常に機能が多く，一度に理解することは難しいと思われます．したがって，必要と思われる機能をその時々にテクニカルマニュアルなどを見て理解する方がよいでしょう．

ここでは次の3つの通信方式について説明します．

①　非同期通信方式

非同期通信はデータビットの前にスタートビットと，後にストップビットが付け加えられています．このため8ビットデータを送信する場合，最

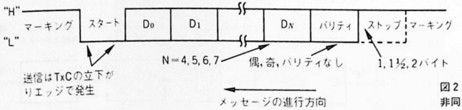


図2・36
非同期通信のデータフォーマット

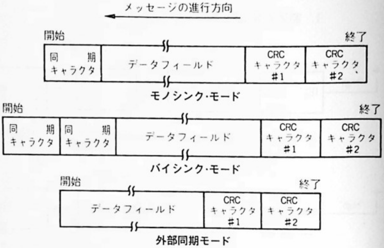


図2・37
同期通信のデータフォーマット

低でも10ビットのデータ長となります(図2・36).

非同期通信ではデータの送信が行なわれていないときには,信号線はマーキングと呼ばれる“High”レベルにあります．そして通信が始まると，まず“Low”レベルのスタートビットが送信され，マーキング状態とは別であることを知らせます．レシーバは“High”から“Low”への変化を検出し，その後決められた時間ごとに信号線からサンプリングしながらデータを入力します．

データはまずLSB（下位バイト）が送信され，MSB（上位バイト）の方向に送信されます．パリティビットを含めて，すべてのデータの送信が終わると，“High”レベルのストップビットが送信されます．

このように非同期通信方式では，各データの先頭にあるスタートビットを検出して，その後のデータをサンプリングします．すなわち，各データごとにサンプリングのタイミングを調整します．したがって，非同期通信方式はデータがいつ送信されるかわからないような通信に用いられます．

②同期通信方式

同期通信は送受信ともに×1のクロックを使用します．データは受信クロック入力の立上がりでサンプリングされます．送信データは送信クロックの立下がりで変化します．

同期式通信方式には同期キャラクタの有無とバイト数により，モノシンク，バイシンク，外部同期という3種類の方式があります．それぞれの方式は第1キャラクタの同期の方法が異なっています（図2・37参照）．

③　SDLC通信方式

SDLC(Synchronous Data Link Control)ではバイシンク方式の同期キャラクタに似たフラグによってスタート，エンドを行います．オープンフラグの後には8ビットのアドレスが続きさらに8ビットの制御フィールドが続きます．クローズビットの前には16ビットの検査データが送信されます．

SDLC通信方式では，制御フィールドを含むデータフィールドはどのようなビットパターンでも送信できるようになっています．また，SDLC通信方式では，送信データは8ビットデータにかぎらずビット単位でも送信可能です(図2・38)．

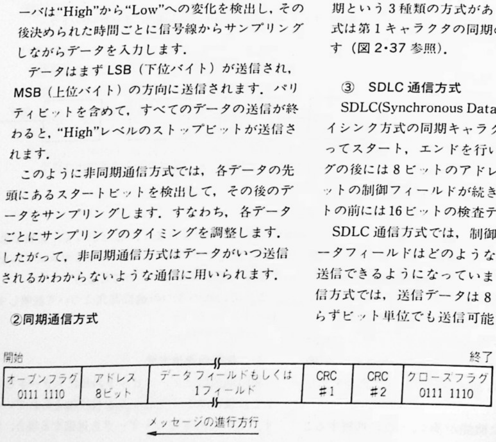


図2・38
SDL通信の
データフォーマット

2-3-6

8255

8255は8ビットマイクロコンピュータ・システム用に開発されたプログラマブル汎用IOデバイスです．8ビットの入出力ポートが3組ありプログラムによるコントロールでデータ入力，出力，ステータス信号入力，出力に使用することができます．

8255のおもな特徴は

① 8085マイクロコンピュータ・コンパチブル
② 各ビットのセット，リセットはCPUから直接制御可能
③ 入出力はTTLコンパチブル
④ パッケージは40ピンDIP
⑤ 4 MHz動作
⑥ 電源は+5 V ±10％

マイクロコンピュータ・システムの周辺装置とのインターフェイスに使用する8255は，システムのプログラムによってコントロールするため外部の論理回路が不要です．

8255は8ビットの入出力ポートが3つあり，ポートA，ポートB，ポートCと呼びます．各ポートごとに独特の性質を持っていますので，ポートをセレクトすることで8255はいろいろ適応することができます．

ポートA，ポートB
入力時：ラッチ，バッファ入力動作
出力時：　〃　出力動作
ポートC
入力時：バッファ入力動作
出力時：ラッチ，バッファ出力動作

このポートCはモード制御によって2つの4ビットポートに分割することができます．

図2·39に8255のピン配置を，表2·9に定格を示します．

8255は次の3つの使用モードに分けられます．

モード0

モード0では基本的な入出力ポートとして動作します．CPUからコントロールワードを入力することによってポートA，ポートB，ポートC（上位），ポートC（下位）の各ポートを入力あるいは出力ポートに設定できます．

モード1

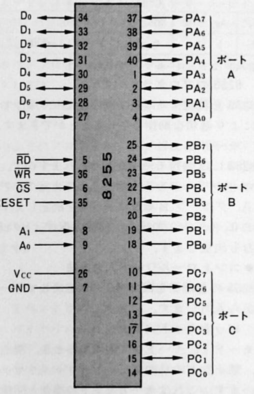


図2·39　8255のピン配置

モード1ではコントロール信号，ステータス信号を用いて，データの入出力制御します．このモードではグループA，グループBを単位とします．

グループA
データポート――ポートA(PA_0～PA_7)
コントロール／ステータスポート
――ポートC(PC_3～PC_7)

グループB
データポート――ポートB(PB_0～PB_7)
コントロール／ステータスポート
――ポートC(PC_0～PC_2)

モード2

モード2ではデータの入力，出力の両方を同一のポートを使用し，双方向バスを通して周辺装置と接続します．データの入出力はモード1と同じくコントロールとステータス信号によって制御します．

項目	記号	定格値	単位
入力電圧	Vin	−0.5～+7.0	V
出力 〃	Vout	−0.5～+7.0	V
動作温度	Topr	0～+70	℃
保存 〃	Tstg	−65～+150	℃

表2·9　8255の最大定格

コントロール・ワード入力時

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	動　作	
1	1	1	0	0	データバス→制御	出力(ライト)

図2・40　コントロールワードの入力セット

8255のプログラミング

8255を使用する場合の順序に従って操作することにより適切な動作をさせることができます.

① モードセット

8255には動作モードが3つありますので，コントロールワードを入力することによって，グループA，グループB単位にモードを決定します．モード0，モード1の場合は同時に各ポートの入力，出力も決定します．

●コントロールワード入力方法

8255の各端子を図2・40の状態にしてデータバスから入力します．

② ビットセット／リセット

モード1，モード2で使用するとき，割込み許可，禁止などの目的でビットセット／リセットを行います．これはモードセットの場合と同様にコントロール・ワードの入力によって行うことができます．

③ データ入力／出力

各ポート（ポートA，ポートB，ポートC）の入力ないし出力データを転送します．モード0の場合はすべてのポートがデータ入力に使用され，データを転送するタイミングはCPUが自由に選ぶことができます．

モード1，モード2の場合8255の動作はコントロール／ステータスポートの状態によって決まります．CPUは割込みがかけられたらその時点でデータの受取りまたは選出をします．

モード1，モード2でポートCを読込むとデータと同様に，ステータスを検出することができます．

●データ入力／出力の方法

8255の各端子を表2・10の状態にしてデータバスを通じて入出力します．

特殊なモード組合せについて

ポートCのすべてのビットが制御またはステータスとして用いられないようないくつかのモードの組合わせがあります．その場合，残りのビットは以下に示すように利用できます．

入力としてプログラムされているとき

すべての入力ラインが通常のポートCリード間にアクセスできます．

出力としてプログラムされるとき

ポートCの下位4ビット(PC_{7-4})は，ビット・セット／リセット機能を用いて，個々にアクセスできます．

ポートCの下位4ビット(PC_{3-0})は，ビット・セット／リセット機能を用いて個々にアクセスできるほかに，下位3ビット(PC_{2-0})は3ビット単位で書込めます．

ポートBとポートCのソース電流能力

ポートBとポートCの中からランダムに選ばれた任意の8本までの出力バッファは，1.5Vで1mAを供給することができます．この特徴によって8255 A-5はダーリントン形ドライバや大きなソース電流を必要とする高電圧ディスプレイをドライブすることができます．

2-3-7 8253

8253は8ビットマイクロコンピュータ・システム用に開発されたプログラマブルカウンタ／タイマです．3組の16ビットカウンタにより構成されており，最大4 MHzのカウントレートで動作します．また，Nチャネル MOS，入出力 TTL コンパチブルであり，+5 V単一電源，24ピンプラスチックDIPです．

表2・10
8255の入出力方法

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	動　作	
0	0	0	1	0	ポートA→データバス	入力(リード)
0	1	0	1	0	ポートB→データバス	
1	0	0	1	0	ポートC→データバス	
0	0	1	0	0	データバス→ポートA	出力(ライト)
0	1	1	0	0	データバス→ポートB	
1	0	1	0	0	データバス→ポートC	

項　目	記 号	定 格 値	単位
入力電圧	Vin	−0.5〜+7.0	V
出力 〃	Vout	−0.5〜+7.0	V
動作温度	Topr	0〜+70	℃
保存 〃	Tstg	−65〜+150	℃

表2・11　8253の最大定格

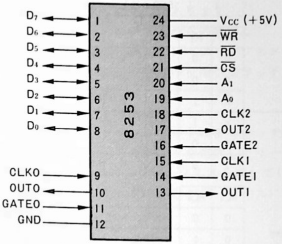


図2・41　8253のピン配置

8253のおもな特徴は

① 8080 A，8085 A コンパチブル
② 3組の16ビットカウンタにより構成されている
③ カウントレートはDC〜最大4 MHz
④ 動作モードは6種類あり，時間設定はプログラムで制御可能
⑤ 2進カウンタとBCDカウンタの選択ができる
⑥ Nチャネル MOS，入出力は TTL コンパチブル
⑦ +5 V 単一電源，24ピン・プラスチック DIP タイプ

8253はプログラム時間設定および6種の動作モードの選択を行ない，マイクロコンピュータシステムにおける広範なタイミング制御に使用することができます．

表2・11に8253の最大定格を，図2・41にそのピン配置を示します．

マイクロコンピュータ・システムでは，CPUと周辺装置間でデータのやりとりをしたり周辺装置をコントロールしたりするような場合では，同期をとるためのタイムディレイが必要となります．通常，これはシステムソフトウェアの中でタイミングループを形成してコントロールしています．8253を用いると次のようにプログラムして同様のタイムディレイを行うことができます．

① CPUの命令により，カウンタをイニシャライズします
② カウンタにより必要なタイムディレイをカウントします
③ カウントが終了するとCPUに割込みをかけます

このようにプログラムすることによって，ソフトウェアのオーバーヘッドが最小となり，また優先レベルの割当てにより多重ディレイが可能となります．

8253ではその他に次のような動作を行うことができます．

1．プログラマブル・レートジェネレータ
2．イベントカウンタ
3．バイナリレート・マルチプライヤ
4．リアルタイムクロック
5．デジタルワンショット
6．コンプレックス・モータコントローラ

8253はマイクロコンピュータ・システムのプログラムによって制御されるため，カウンタを動作させる場合は，コントロールワードとカウント数をあらかじめ指定する必要があります．プログラムはカウンタ別に付属するモードレジスタ，カウンタレジスタにロードします．一度プログラムしますと新たにコントロールワードを書き込まない限り，その指定したモードで動作します．

カウントが終了するとOUT端子に結果が出力されますがカウント中のデータを読み出すこともできます．

8253のプログラミングを図2・42にもとづいて説明しましょう．

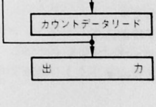


図2・42
8253の動作フロー

D7	D6	D5	D4	D3	D2	D1	D0
SC_1	SC_0	RL_1	RL_0	M_2	M_1	M_0	BCD

BCD	
0	バイナリカウント（16桁）
1	BCDカウント（4桁）

M_2	M_1	M_0	
0	0	0	モード0
0	0	1	モード1
×	1	0	モード2
×	1	1	モード3
1	0	0	モード4
1	0	1	モード5

RL_1	RL_0	
0	0	カウント・ラッチ動作
0	1	LSBのリード/ロード
1	0	MSBのリード/ロード
1	1	LSB, MSBの順にリード/ロード

SC_1	SC_0	
0	0	カウンタ #0
0	1	カウンタ #1
1	0	カウンタ #2
1	1	―――　(注)

(注) SC_1, SC_0=1, 1は使用できない．×：Don't care

図2・43　コントロールワードのフォーマット

モード指定はコントロールワードで行ないます．コントロールワードは8ビットより成り図2・43のように4ブロックに分けられます．

① SC_1～SC_0

カウンタ#0，#1，#2のうちの1つをセレクトします．以降②～④はこれでセレクトしたカウンタに対して行なわれます．

② RL_1～RL_0

カウントレジスタにロードするカウント数およびリードするカウントデータのバイト長（1 or 2バイト）またはカウントデータラッチ動作の指定を行います．

③ M_2～M_0

モード0～5を指定します．

モード0：カウント終了時割込み
1：プログラマブルワンショット
2：レートジェネレータ
3：方形波レートジェネレータ
4：ソフトウェア・トリガストローブ
5：ハードウェア・トリガストローブ

④ BCD

バイナリカウントおよびBCDカウントのセレクトを行います．

モード指定後，そのカウンタに対しカウント数を指定します．カウンタ数のバイト長はコントロールワードで指定した$RL_{1\sim0}$に従います．バイト長が2バイトのときは下位，上位の順にロードします．

カウント数の指定が終わり，CLK 入力の立上り，立下がりの連続があると，立下がりでカウントレジスタにセットされているカウント数がカウンタにセットされ，カウントを開始します．

以上がカウンタを動作させるための書込み手順です．

また，カウンタを使用するアプリケーションでは，カウント中にその値を読み出し，それに基づいて処理を行う場合もあります．8253 ではこのカウント読出し機能があり，カウント中のカウンタに影響することなしでカウンタの内容を読出すことができます．読出される内容はモード指定で $RL_{1\sim0}$ で指定したバイトです．

読出し動作には次の２つがあります．

Ⓐ　**ストレージ・レジスタからの読出し**

カウンタの内容を一時ストレージ・レジスタにロードし，それからリード動作を用いてラッチされているデータを読み出します．

Ⓑ　**カウンタからの直接読出し**

Ⓐのようにコントロールワードによりラッチ動作を行なわず，$A_{1\sim0}$ でカウンタを選択してリード動作を行うと，ストレージ・レジスタを通して(ラッチされない）カウンタの内容がデータバスに読み出されます．

2-3-8 CRT，メモリコントローラ

CRTコントローラ

CRT コントローラには，その機能によって次のように分けられます．

① CRTC（6845 など）

機能：表示アドレスおよび，水平，垂直同期信号を発生する

② VDP（9918 など）

機能：スプライト機能など，特徴的な表示機能を持つもの

③ GDC（7220 など）

機能：直線を描く，スクロールを行うといった，描画機能を持ったコントローラで，ハードウェアによる描画が可能であるため，高速動画が可能となる

MZ-1500 の場合には，専用の IC を採用していますが，機能としては上記分類の①に相当します．

CRTC は，パソコンの顔といえるものです．MZ-1500 の CRTC は，PCG をサポートしており，リアルタイムの動画を実現できます．PCG は文字を表示する CG が通常 ROM で構成されているのを RAM に置き換えを行ない，任意に CG コードの書換えを可能としたシステムです．

図 2･44 に CRTC の基本ブロックを示します．

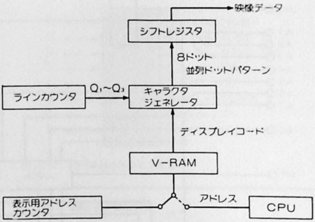


図 2･44　CRTC の基本ブロック図

メモリコントローラ

メモリコントローラは文字通り，CPU に接続しているメモリを管理する回路です．MZ-1500 のメモリコントローラは，CRTC と同一のチップに入っており，次の機能を持っています．

① DRAM のリフレッシュなどの制御

② VRAM，およびメインメモリのバンク管理

③ 各周辺 IC のチップセレクト管理

これらの機能は図 2･45 のブロックのように構成されています．同図で

$\overline{RAS}$：DRAM の RAS 信号

$\overline{CSO}$：ROM の CS

$\overline{WA}$：CPU への WAIT

$\overline{CSE}$：メモリマップド I／O の CS

$\overline{PRC}$：プリンタバスの CS

$\overline{CSDD}$：VRAM セレクト信号

INH1~INH3：各バンクセレクト

MZ-1500 でのメモリ管理は図 2･46 のようになっており，電源 ON 時およびリセットスイッチ ON 時には図 2･47 のようになっています．

2-3-9 PSG

SN 76489 は低価格のトーン，ノイズ発生器でマイクロコンピュータに最適であり，データバスをベースとした IO 周辺デバイスとして開発されたものです．

標準的な CPU と簡単な周辺ロジックをそなえることにより，ミュージック演奏や擬音の発生をさせることができます．

この SN 76489 を使うことによって，従来のアナログ回路による方式に比べハード構成が大幅に簡

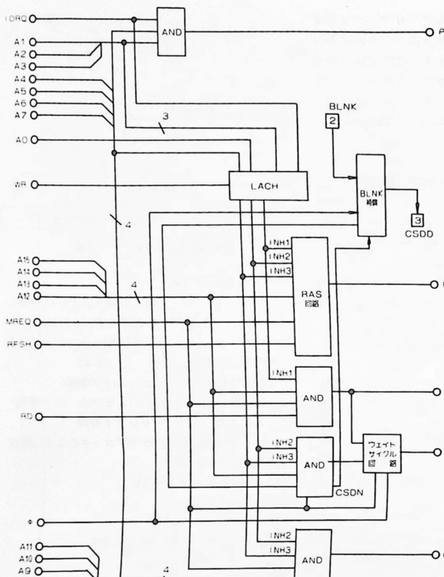


図2・45
メモリコントロールブロック

略化できます．ソフトウェアでの管理が可能なため，サウンドパターンを開発するごとに ROM などにプログラムしておくことによってサウンドファイルの切換えやサイクル的な動作が容易に得られます．SN 76489 の応用として，自動演奏マシン，ゲームマシンの効果音等が考えられます．

SN 76489 のおもな特徴は

① 8ビット並列データバスによりプログラム可能な3個のトーン発生器を内蔵

② 8ビット並列データバスによりプログラム可能な1個のノイズ発生器を内蔵

③ 8ビット並列データバスによりプログラム可能な減衰器を内蔵

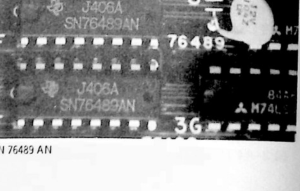
④ 3.579 MHz のリファレンスクロックを使用

⑤ 3和音の出力が可能

⑥ 数 Hz～クロック／32 と，出力周波数範囲が広い

図 2・48 に SN 76489 AN のピン配置を示します．

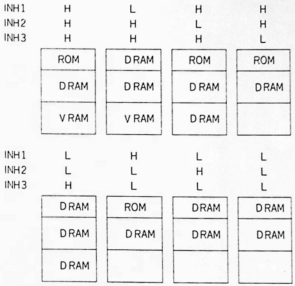
なお PSG の具体的な内容は 3 章で触れることにします．



SN 76489 AN

I/O ポート	出 力 データ	$0000 ～$0FFF	$D000 ～$FFFF
$E0	—	DRAM	—
$E1	—	—	DRAM
$E2	—	モニタ ROM	—
$E3	—	—	VRAM8255, 8253 MZ−1500 ROM
$E4	—	モニタ ROM —	V−RAM8255, 8253 MZ−1500 ROM
$E5	00	—	CG−ROM
	01	—	PCG RAM ˙BLUE
	02	—	PCG RAM ˙RED
	03	—	PCG RAM ˙Green
$E6	—	—	$E5を実行する前の状態に戻る

INH1～INH3は，カスタムLSI内部の信号で，この信号により，メモリマップが変化する．



注）メモリ切換を行う命令をその切換えられるメモリブロック内に書いてはいけない．

図2・46 メモリ管理

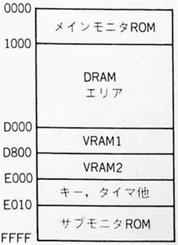


図2・47 電源ON時のメモリマップ

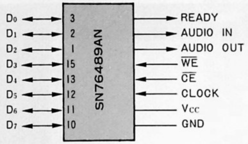
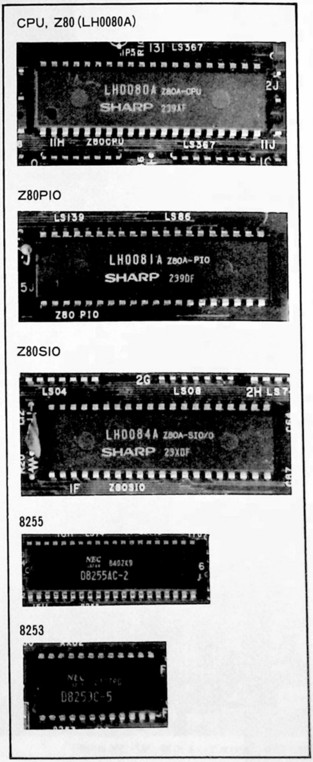
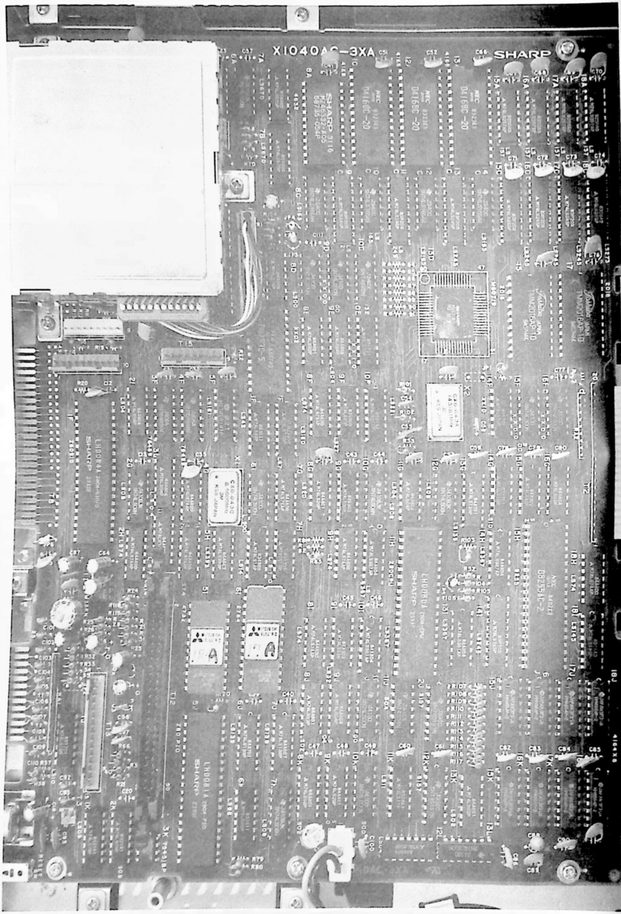


図2・48 SN 76489 ANのピン配置





MZ-1500に使われているIC群（MZ-1500の内部）

第3章 ハードウェア

3-1：MZ-1500のソフトウェア

BASICについて
IPLについて
モニタについて

3-2：MZ-1500のハードウェア

MZ-1500の構成
RAMファイル
CRTC
QDとQDコントローラ
QDハンドラー
カセットインターフェイス
キーボードスキャン
プリンタインターフェイス
PSG

3-1 MZ-1500のソフトウェア

3-1-1 BASICについて

MZ-1500のBASIC(1500-BASIC)は，MZ-700シリーズのS-BASICをもとに，グラフィックスを中心に拡張したものです．

従って，S-BASICに対しては，ほとんど互換性がありますが一部互換性のないコマンドもあります．ここでは互換性のない部分に焦点をあててまとめてみることにします．

S-BASICのプログラムを移植するときはこれらの項目に注意が必要となります．

① コマンド名の変更

下のように変更されています．

ただし，中間コードは変更していないのでS-BASICで作られ，テープまたはフロッピーディスクにSAVEされたプログラムは，1500のBASICでLOADすれば自動的にこの変更が行われるので，特別に注意する必要はありません．

問題になるのはプログラムリストに従ってキーボードから入力する場合です．このときは表のとおりに変更しながら入力する必要があります．

S-BASIC	1500-BASIC
COLOR	CCOLOR
SET	CSET
RESET	CRESET
LINE	PLINE
CIRCLE	PCIRCLE
MODE	PMODE
MOVE	PMOVE
TEST	PTEST
SKIP	PSKIP
INP #	INP @
OUT #	OUT @

② テープに関するコマンド

1500-BASICでは，ファイル関係が複雑化するのを避けるため，ファイルディスクリプタを採用しており，テープのディスクリプタは CMT: です．従ってテープに関するコマンドが使用されているプログラムは移植時，次の対応が必要になります．

●先頭に次の命令を書く

機　種		カセット	QD	DISK
MZ-700		36442	30142	30406
MZ-1500		—	22944	22939
〃	(NEW ON 1)	—	24760	22939
〃	(NEW ON 2)	—	25843	22939

表3・1　フリーエリアの比較　(単位はバイト)

DEFAULT　CMT:

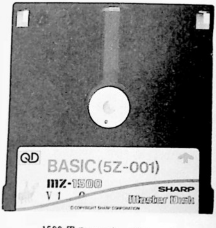
●グローバルスイッチ(／T)がついているコマンドがあれば，グローバルスイッチ(／T)を削除．

③ フリーエリア

1500-BASICは，S-BASICに比べ容量が大きく，その分フリーエリアが狭くなっています．従って大きな容量のプログラムや，大量に変数や機械語エリアを確保するプログラムは，移植できない可能性があります．この場合，プログラムを分割してSAVEし，CHAIN文でつなぐなどの対応をケースバイケースでしなければいけません．

それぞれの場合のフリーエリアを比較したのが表3・1です．

最後に1500-BASIC(DISK-BASICを含む)の全コマンドについてS-BASICと比較(S-BASICの部分と拡張部分）した表を掲載します．中間コード表は前章も参照して下さい．中間コード表に記載してあるコマンドは一部説明していないものもありますが，すべて予約語になります（表3・2A, B参照)．



1500用のマスターQD

MZ−700 S−BASIC コマンド一覧表			
SYSTEM	FILE	CRT/GRAPHIC	KEY
AUTO	*SAVE	CONSOLE	DEF KEY
BYE	*LOAD	*CCOLOR	GET
CLR	*MERGE	CURSOR	INPUT
CONT	VERIFY	PRINT	
DEF FN		PRINT USING	
*DELETE		PRINT 〔a,b〕	
END		*CSET/CRESET	
FOR…NEXT		SPC	
GOTO		TAB	
GOSUB…RETURN			
IF…ERL			
…ERN			
…GOTO			
…GOSUB			
* …THEN…ELSE			
KEYLIST			
LET			
LIST(/P)			
NEW			
ON…ERROR…GOTO			
…GOTO			
…GOSUB			
READ…DATA			
REM			
RENUM			
RESTORE			
RESUME			
RUN			
SIZE			
STOP			

*印は、名称変更・機能拡張を示します
なお、1500−BASIC用に追加されたコマンドについては、後述参照ください

表3・2 A　MZ-700, MZ-1500 の BASIC コマンド一覧

MZ−700　S−BASIC　コマンド一覧表			
関数系	PRINTER	SOUND/TIMER	機械語
ABS ASC ATN CHR$ COS DIM EXP INT LEFT$ LEN LN LOG MID$ PAI RAD RIGHT$ RND SNG SIN SIZE SQR STR$ TAN VAL	AXIS GPRINT HSET *PMODE…GR …TN …TL …TS *PMOVE PAGE *PCOLOR *PCIRCLE *PLINE PHOME PLOT ON/OFF RLINE RMOVE PSKIP PTEST LIST/P	*MUSIC TEMPO TI$	INP@ OUT@ PEEK POKE *USR LIMIT

*印は，名称変更・機能拡張を示します
なお，1500−BASIC用に追加されたコマンドについては，後述参照ください

表3・2A

1500−BASIC　新コマンド一覧表			
SYSTEM	FILE	CRT/GRAPHIC	KEY
SEARCH(/P) BOOT LABEL TRON/TROFF NEWON	INIT DEFAULT DIR(/P) ★LOCK/UNLOCK ROPEN#/WOPEN# ★XOPEN# INPUT#/PRINT# KILL#/CLOSE# ★DELETE/RENAME CHAIN ★SWAP	COLOR CLS SET/RESET LINE/BLINE PAL CIRCLE PAINT BOX POSITION PATTERN FPRINT	

★印は，DISK−BASIC専用のコマンドです
なお，★印のコマンドを含め，FILE関係のコマンドは，MZ−700のDISK−BASICですでに動作しています

表3・2A

1500−BASIC　新コマンド一覧表			
関数系	PRINTER	SOUND/TIMER	機械語
EOF CSRH/CSRV POSH/POSV HEXCHR$ ASCCHR$ POINT NOT AND OR XOR	HCOPY DIR/P SEARCH/P TRON/P	SOUND NOISE MUSIC　WAIT STOP	

表3・2 A

	00	10	80	90	A0	B0	C0	D0	E0	F0
0	(eol)		GOTO		POKE	HSET		ROPEN	TO	=<
1		16進数	GOSUB	INPUT	OFF	GPRINT	ERROR		STEP	<=
2					PMODE	KEY	ELSE		THEN	=>
3			RUN	IF	PSKIP	AXIS	USR		USING	>=
4			RETURN	DATA	PLOT	LOAD	BYE		n	=
5		実数	RESTORE	READ	PLINE	SAVE		DIR	ALL	>
6			RESUME	DIM	PLINE	MERGE			TAB	<
7			LIST	REM	PMOVE	CHAIN	DEF		SPC	+
8				END	RMOVE	CONSOLE		RENAME		−
9			DELETE	STOP	TRON	SEARCH		KILL		¥
A			RENUM	CONT	TROFF	OUT@	LABEL	LOCK	XOR	MOD
B	行番号		AUTO	CLS	INP@	PCIRCLE		UNLOCK	OR	/
C	アドレス		EDIT		DEFAULT	PTEST		INIT	AND	*
D			FOR	ON	GET	PAGE			NOT	∧
E			NEXT	LET	PCOLOR	WAIT	WOPEN		<>	
F			PRINT	NEW	PHOME	SWAP	CLOSE		∧	

	FE80	FE90	FEA0	FF80	FF90	FFA0	FFB0	FFC0
0		COLOR		INT		CHR$		
1	CSET	PRTY		ABS		STR$		
2	CREST	SET	MUSIC	SIN		HEX$	HEXCHR$	
3	CCOLOR	RESET	TEMPO	COS			ERN	
4		LINE	CURSOR	TAN			ERL	TI$
5		BLINE	VERIFY	IN			SIZE	POINT
6		PAL	CLR	EXP			SCRH	EOF
7		CIRCLE	LIMIT	SQR			SCRV	FN
8		BOX		RAD		SPACE$	POSH	
9		PAINT		PEEK			POSV	
A	SOUND	POSITION		ATN		ASCCHR$	LEFT$	
B		PATTERN		SGN		ASC	RIGHT$	
C	NOISE	HCOPY		LOG		LEN	MID$	
D	BEEP			FRAC		VAL	FONT$	
E	VOICE	FPRINT	BOOT	PAI	JOY			
F		KPATTERN		RAD				

表3・2 B
BASIC中間コード表

3-1-2

IPLについて

MZ-1500のIPLに関して，リストを掲載します．この中にはカセットテープ，QDの読み書きなど有用なルーチンが含まれておりますので解析して利用してください．

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 01               84.06.26

E800                      ORG     E800H
E800                  ;================================
E800                  ;
E800                  ;          EQUATE  FILE
E800                  ;             for MZ-1500
E800                  ;             1984.5.09
E800                  ;
E800                  ;================================
E800                  ;
E800                  ;
E800                  ; MONITOR EQU TABLE
E800                  ;
10F0  P               SP:       EQU     10F0H
003E  P               BELL:     EQU     003EH
0006  P               LETNL:    EQU     0006H
000C  P               PRNTS:    EQU     000CH
0012  P               PRNT:     EQU     0012H
0015  P               MSG:      EQU     0015H
001E  P               BRKEY:    EQU     001EH
0003  P               GETL:     EQU     0003H
001B  P               GETKY:    EQU     001BH
0DDC  P               ?DPCT:    EQU     0DDCH
0BB9  P               ?ADCN:    EQU     0BB9H
0009  P               NL:       EQU     0009H
03BA  P               PRTHL:    EQU     03BAH
03DA  P               ASC:      EQU     03DAH
0410  P               HLHEX:    EQU     0410H
041F  P               2HEX:     EQU     041FH
04D8  P               ?RDI:     EQU     04D8H
10F1  P               NAME:     EQU     10F1H
04F8  P               ?RDD:     EQU     04F8H
10F0  P               ATRB:     EQU     10F0H
1102  P               SIZE:     EQU     1102H
1104  P               DTADR:    EQU     1104H
1106  P               EXADR:    EQU     1106H
0027  P               RDINF:    EQU     0027H
002A  P               RDDAT:    EQU     002AH
0470  P               LLPT:     EQU     0470H
03D5  P               SLPT:     EQU     03D5H
05FA  P               NLPHL:    EQU     05FAH
03B1  P               SPHEX:    EQU     03B1H
0920  P               ?PRTS:    EQU     0920H
0126  P               BGETL:    EQU     0126H
02A6  P               .4DE:     EQU     02A6H
00EA  P               GOTO:     EQU     00EAH
0436  P               ?WRI:     EQU     0436H
0475  P               ?WRD:     EQU     0475H
0942  P               MSGOK:    EQU     0942H
0588  P               ?VRFY:    EQU     0588H
005B  P               CMYO:     EQU     005BH
119D  P               SWRK:     EQU     119DH
1171  P               DSPXY:    EQU     1171H
E000  P               KEYPA:    EQU     E000H
E001  P               KEYPB:    EQU     E001H
08CA  P               ?KEY:     EQU     08CAH
0A32  P               ?BRK:     EQU     0A32H
0147  P               MSGE1:    EQU     0147H
096C  P               PRNT3:    EQU     096CH
0DA6  P               ?BLNK:    EQU     0DA6H
06E7  P               MSG?3:    EQU     06E7H
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 02            84.06.26

001E  P         BRKCK:   EQU   001EH
0FD8  P         ?CLER:   EQU   0FD8H
038D  P         TIMIN:   EQU   038DH
119E  P         TEMPW:   EQU   119EH
02BE  P         MLDSP:   EQU   02BEH
0577  P         ?BEL:    EQU   0577H
073E  P         ?MODE:   EQU   073EH
09DE  P         #CLR8:   EQU   09DEH
0308  P         ?TMST:   EQU   0308H
E800            ;
E800            ;
E800            ; QUICK DISK EQU TABLE
E800            ;
E800            ;
00F4  P         SIOAD:   EQU   F4H          ;SIO A DATA
00F5  P         SIOBD:   EQU   F5H          ;SIO B DATA
00F6  P         SIOAC:   EQU   F6H          ;SIO A CONTROL
00F7  P         SIOBC:   EQU   F7H          ;SIO B CONTROL
1130  P         QDTBL:   EQU   1130H        ;IOCS PARAMETER ADRS
1130  P         QDPA:    EQU   QDTBL        ;IOCS COMMAND
1131  P         QDPB:    EQU   QDPA+1       ;IOCS PARAMETER
1132  P         QDPC:    EQU   QDPB+1       ;DATA HEAD ADRS(1)
1134  P         QDPE:    EQU   QDPC+2       ;DATA BYTE SIZE(1)
1136  P         QDPG:    EQU   QDPE+2       ;DATA HEAD ADRS(2)
1138  P         QDPI:    EQU   QDPG+2       ;DATA BYTE SIZE(2)
113A  P         QDCPA:   EQU   QDPI+2       ;QD FLAG BYTE (A)
113B  P         QDCPB:   EQU   QDCPA+1      ;QD FLAG BYTE (B)
113C  P         QDCPC:   EQU   QDCPB+1      ;QD FLAG BYTE (C)
113D  P         HDPT:    EQU   QDCPC+1      ;IOCS WORK
113E  P         HDPTO:   EQU   HDPT+1
113F  P         FNUPS:   EQU   HDPTO+1
1140  P         FNUPS1:  EQU   FNUPS+1
1141  P         FNUPF:   EQU   FNUPS1+1
1142  P         FNA:     EQU   FNUPF+1
1143  P         FNB:     EQU   FNA+1
1144  P         MTF:     EQU   FNB+1
1145  P         RTYF:    EQU   MTF+1
1146  P         BFRF:    EQU   RTYF+1
1147  P         SYNCF:   EQU   BFRF+1
1148  P         RETSP:   EQU   SYNCF+1
1104  P         QSIZE:   EQU   1104H        ;QD FILE SIZE
1106  P         QDTADR:  EQU   1106H        ;QD DATA ADDR
1108  P         QEXADR:  EQU   1108H        ;QD EXEC ADDR
0011  P         NAMSIZ:  EQU   0011H        ;NAME SIZE
CD90  P         QDIRBF:  EQU   CD90H        ;QD DIR BUFFER
0001  P         OBJCD:   EQU   01H          ;OBJ CORD
0002  P         BTXCD:   EQU   02H          ;BTX CORD
0003  P         BSDCD:   EQU   03H          ;BSD CORD
0000  P         BRKCD:   EQU   00H          ;BREAK CODE
0028  P         NTFECD:  EQU   40           ;NOT FOUND ERR CORD
0029  P         HDERCD:  EQU   41           ;HARD ERR CORD
002A  P         ALEXCD:  EQU   42           ;ALREADY EXIST CORD
002E  P         WPRTCD:  EQU   46           ;WRITE PROTECT ERR CORD
0032  P         QNTRCD:  EQU   50           ;NOT READY
0033  P         TMFECD:  EQU   51           ;TOO MANY FILES ERR
0035  P         NFSECD:  EQU   53           ;NO FILE SPACE ERR CORD
0036  P         UNFMCD:  EQU   54           ;UNFORMAT ERR CORD
0039  P         BDSKCD:  EQU   57           ;BAD DISK ERR CORD
E800            ;
E800                     SKP   H
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 03                    84.06.26

E800                  ;
E800                  ;     MZ-1500 INITIAL PROGRAM LOADER
E800                  ;
E800                  ;             & SUBMONITOR (9Z-502M V1.0C)
E800                  ;
E800                  ;                                 1984.05.09
E800                  ;
E800                  ;
E800                  ;
11A3  P               BUFER:  EQU     11A3H           ;KEYIN BUFFER
E800                  ;
E800                  ;
E800 00               START0: NOP
E801 C310EB                   JP      STARTX          ;SUB MONITOR START
E804                  STRT00: ENT
E804 C3A7EA                   JP      ST1X            ;COMMAND
E807                  PSGRS0: ENT
E807 C32FE8                   JP      PSGRS           ;PSG RESET
E80A                  PALRES: ENT
E80A C381E8                   JP      PALT            ;PALLET RESET
E80D                  QDIOS:  ENT
E80D C300FA                   JP      QDIO
E810                  ;
E810                  ;
E810                  ;================================
E810                  ;
E810                  ;       SUB MONITOR START
E810                  ;
E810                  ;================================
E810                  ;
E810                  ;
E810                  STARTX: ENT
E810                  ;
E810                  ;
E810 31F010                   LD      SP,SP           ;STACK SET (10F0H)
E813 ED56                     IM      1               ;IM 1 SET
E815 CD3E07                   CALL    ?MODE           ;8255 MODE SET,INT SET
E818 AF                       XOR     A               ;8253 MODE SET
E819 110000                   LD      DE,0000H        ;
E81C CD0803                   CALL    ?TMST           ;
E81F F3                       DI                      ;
E820 CD320A                   CALL    ?BRK            ;CTRL ?
E823 3005                     JR      NC,PSGRES
E825 FE20                     CP      20H             ;KEY IS CTRL KEY
E827 CA5B00                   JP      Z,CMYO          ;MEMORY CHENGE
E82A                  ;
E82A                  ;    PSG RESET
E82A                  ;
E82A CD2FE8           PSGRES: CALL    PSGRS
E82D 180B                     JR      SD&VC
E82F 0604             PSGRS:  LD      B,04H
E831 3E9F                     LD      A,9FH
E833 D3E9             PSGSET: OUT     (E9H),A
E835 C620                     ADD     A,20H
E837 10FA                     DJNZ    PSGSET
E839 C9                       RET
E83A                  ;
E83A                  ; SOUND MASK & VOICE RESET
E83A                  ;
E83A 3E01             SD&VC:  LD      A,01H
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 04           84.06.26

E83C 3203E0                LD      (E003H),A       ;SOUND MASK RESET
E83F 3E80                  LD      A,80H
E841 D3E8                  OUT     (E8H),A         ;VOICE INIT.
E843               ;
E843               ;
E843               STOX:   ENT
E843 06FF                  LD      B,FFH           ; BUFFER CLEAR
E845 21F110                LD      HL,NAME         ; 10F1H-11F0H CLEAR
E848 CDD80F                CALL    ?CLER
E84B 3E16                  LD      A,16H           ; LASTER CLR
E84D CD1200                CALL    PRNT
E850 3E71                  LD      A,71H           ; BACK:BLUE  CHARA:WHITE
E852 2100D8                LD      HL,D800H        ; COLOR ADRS
E855 CDDE09                CALL    #CLR8
E858 218D03                LD      HL,TIMIN        ;INTERRUPT JUMP ROUTINE
E85B 3EC3                  LD      A,C3H           ;
E85D 323810                LD      (1038H),A
E860 223910                LD      (1039H),HL
E863 3E04                  LD      A,04H           ;NORMAL TEMPO
E865 329E11                LD      (TEMPW),A
E868 CDBE02                CALL    MLDSP           ;MELODY STOP
E86B CD0900                CALL    NL
E86E CD7705                CALL    ?BEL
E871               SSX:    ENT
E871 3E01                  LD      A,01H
E873 329D11                LD      (SWRK),A        ;KEY IN SILENT
E876               ;
E876               ; PALLET & PRIORITY RESET
E876               ;
E876 CDA1EA                CALL    DISCLR
E879 CD81E8                CALL    PALT            ;PALLET RESET
E87C AF                    XOR     A
E87D D3F0                  OUT     (F0H),A         ;PRIORITY RESET (CG>PCG)
E87F 180C                  JR      PCGRES
E881 0EF1          PALT:   LD      C,F1H
E883 AF                    XOR     A
E884 0608                  LD      B,08H
E886 ED79          PALLET: OUT     (C),A
E888 C611                  ADD     A,11H
E88A 10FA                  DJNZ    PALLET
E88C C9                    RET
E88D               ;
E88D               ; PCG RESET
E88D               ;
E88D 219BE8        PCGRES: LD      HL,PCGC
E890 1100C0                LD      DE,C000H
E893 013000                LD      BC,0030H
E896 EDB0                  LDIR
E898 C300C0                JP      C000H
E89B 3E01          PCGC:   LD      A,01H
E89D 0603                  LD      B,03H
E89F D3E5          PCGCL:  OUT     (E5H),A         ;BANK
E8A1 3C                    INC     A
E8A2 1807                  JR      PCGCLR
E8A4 10F9          PCGCLO: DJNZ    PCGCL
E8A6 D3E6                  OUT     (E6H),A
E8A8 C3E0E8                JP      PIOINI          ;PIO INITIALIZE
E8AB F5            PCGCLR: PUSH    AF
E8AC C5                    PUSH    BC
E8AD AF                    XOR     A
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 05                  84.06.26

E8AE 2100D0                  LD     HL,D000H
E8B1 77                      LD     (HL),A
E8B2 1101D0                  LD     DE,D001H
E8B5 01FF1F                  LD     BC,1FFFH
E8B8 EDB0                    LDIR
E8BA C1                      POP    BC
E8BB F1                      POP    AF
E8BC 18E6                    JR     PCGCLO
E8BE                ;
E8BE                ; Z80 PIO INITIALIZE
E8BE                ;
E8BE 0EFC           PIOA:    LD     C,FCH             ;PIO PORT=A
E8C0 AF                      XOR    A                 ;INT VECTOR
E8C1 ED79                    OUT    (C),A
E8C3 3ECF                    LD     A,CFH             ;MODE 3 (BIT)
E8C5 ED79                    OUT    (C),A
E8C7 3E3F                    LD     A,3FH             ;I/O REG SET
E8C9 ED79                    OUT    (C),A
E8CB 3E07                    LD     A,07H             ;INT SEQENS
E8CD ED79                    OUT    (C),A
E8CF 0EFD           PIOB:    LD     C,FDH             ;PIO PORT=B
E8D1 AF                      XOR    A                 ;INT VECTOR
E8D2 ED79                    OUT    (C),A
E8D4 3ECF                    LD     A,CFH             ;MODE 3 (BIT)
E8D6 ED79                    OUT    (C),A
E8D8 AF                      XOR    A                 ;I/O REG SET
E8D9 ED79                    OUT    (C),A
E8DB 3E07                    LD     A,07H             ;INT SEQENS
E8DD ED79                    OUT    (C),A
E8DF C9                      RET
E8E0 CDBEE8         PIOINI:  CALL   PIOA
E8E3                ;
E8E3                ; SIO INITIALIZE
E8E3                ;
E8E3 3E01                    LD     A,01H
E8E5 D3F7                    OUT    (SIOBC),A
E8E7 AF                      XOR    A
E8E8 D3F7                    OUT    (SIOBC),A         ;WR1 REG. CLEAR
E8EA                ;
E8EA                ; MODE CHECK & JUMP
E8EA                ;
E8EA 3AFFFF                  LD     A,(FFFFH)
E8ED FEA5                    CP     A5H
E8EF 2807                    JR     Z,KEYCK
E8F1 11E706                  LD     DE,MSG?3          ;'*** MONITOR ***'
E8F4 DF                      RST    3
E8F5 C3AD00                  JP     00ADH
E8F8                ;
E8F8                ; KEY CHECK
E8F8                ;
E8F8 CD1B00         KEYCK:   CALL   GETKY             ;KEY CHECK
E8FB FE4D                    CP     'M'
E8FD CA6AE9                  JP     Z,MON
E900 FE51                    CP     'Q'
E902 CA0EEA                  JP     Z,QBT
E905 FE43                    CP     'C'
E907 CA74E9                  JP     Z,CSS
E90A                ;
E90A                ; FLOPPY CHECK & BOOT
E90A                ;
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 06          84.06.26

E90A CD18E9                     CALL    FDCC&               ;FD I/O CHECK
E90D 2015                       JR      NZ,R12C&B
E90F CDA1EA           FDB:      CALL    DISCLR
E912 CD7FF4                     CALL    FDX
E915 C37BEA                     JP      ERRMG
E918                  FDCC&:    ENT
E918 3EA5                       LD      A,A5H
E91A 47                         LD      B,A
E91B D3D9                       OUT     (D9H),A
E91D CD14F7                     CALL    DLY8OU
E920 DBD9                       IN      A,(D9H)
E922 B8                         CP      B
E923 C9                         RET
E924                  ;
E924                  ; MZ-1R12 CHECK & BOOT
E924                  ;
E924 0EF8             R12C&B:   LD      C,F8H
E926 CD2EF8                     CALL    R12CK1              ;SET...CY='0',Z='1'
E929 3008                       JR      NC,R12B0
E92B 0EA8                       LD      C,A8H
E92D CD2EF8                     CALL    R12CK1
E930 DA0EEA                     JP      C,QBT
E933 CD34F7           R12B0:    CALL    1R12R0
E936 2001                       JR      NZ,R12B1
E938 E9                         JP      (HL)
E939 119AEE           R12B1:    LD      DE,12ERMG           ;'SRAM:Check sum err'
E93C C37BEA                     JP      ERRMG
E93F                  ;
E93F                  ; DEVICE SELECTION
E93F                  ;
E93F CD8AEA           SLMSG:    CALL    CLLET
E942 CD56EA           SELMG0:   CALL    SELMSG              ;SELECTION MESSAGE
E945 CD20F2                     CALL    NKYWT               ;WAIT NON KEY
E948 3E02             KSCAN0:   LD      A,02H
E94A CDA1F1                     CALL    SRCK0               ;KEY SCAN
E94D FE7F                       CP      7FH                 ;'Q'
E94F CA0EEA                     JP      Z,QBT
E952 3E04                       LD      A,04H
E954 CDA1F1                     CALL    SRCK0
E957 FEDF                       CP      DFH                 ;'C'
E959 CA74E9                     JP      Z,CSS
E95C FEFB                       CP      FBH                 ;'F'
E95E CA0FE9                     JP      Z,FDB
E961 3E03                       LD      A,03H
E963 CDA1F1                     CALL    SRCK0
E966 FEF7                       CP      F7H                 ;'M'
E968 20DE                       JR      NZ,KSCAN0
E96A                  ;
E96A                  ; MONITOR JUMP
E96A                  ;
E96A CDA1EA           MON:      CALL    DISCLR
E96D 11AFEE                     LD      DE,MONMSG           ;'**  MONITOR  **'
E970 DF                         RST     3
E971 C3A7EA                     JP      ST1X
E974                  ;
E974                  ;========================
E974                  ;
E974                  ;     CASSETTE BOOT-UP
E974                  ;
E974                  ;========================
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 07                 84.06.26

E974                     ;
E974                     CSS:    ENT
E974                     ;
E974                     ; MOTOR CHECK
E974                     ;
E974 2102E0                      LD      HL,E002H
E977 7E                  WMODO:  LD      A,(HL)
E978 E610                        AND     10H
E97A 2025                        JR      NZ,DSILF
E97C 23                          INC     HL                  ;PC3:"L"→"H"
E97D 3E06                        LD      A,06H               ;
E97F 77                          LD      (HL),A              ;
E980 3C                          INC     A                   ;Acc←07
E981 77                          LD      (HL),A              ;
E982 2B                          DEC     HL                  ;
E983 7E                          LD      A,(HL)
E984 E610                        AND     10H
E986 2019                        JR      NZ,DSILF
E988                     ;
E988                     ; DISP 'Make ready CMT'
E988                     ;
E988 CDA1EA                      CALL    DISCLR
E98B CD0600                      CALL    LETNL
E98E CD0600                      CALL    LETNL
E991 11E6ED                      LD      DE,IPLM1            ;'Make ready CMT'
E994 CD95EA                      CALL    MSGLET
E997                     ;
E997                     ; WAIT MOTOR ON
E997                     ;
E997 CD1E00              WMOD1:  CALL    BRKEY               ;BREAK ?
E99A 2867                        JR      Z,TPERX0
E99C 7E                          LD      A,(HL)
E99D E610                        AND     10H
E99F 28F6                        JR      Z,WMOD1
E9A1                     ;
E9A1                     ; DISP 'IPL is looking for'
E9A1                     ;
E9A1 CDA1EA              DSILF:  CALL    DISCLR
E9A4 CD0600                      CALL    LETNL
E9A7 1117EE                      LD      DE,IPLM4            ;'IPL is looking ...'
E9AA DF                          RST     3
E9AB                     ;
E9AB                     ; READ INFORMATION BLOCK
E9AB                     ;
E9AB CD2700                      CALL    RDINF
E9AE DA01EA                      JP      C,TPERX
E9B1                     ;
E9B1                     ; DISP 'IPL is loading...'
E9B1                     ;
E9B1 CDA1EA                      CALL    DISCLR
E9B4 11D4ED                      LD      DE,IPLM0            ;'IPL is loading '
E9B7 DF                          RST     3
E9B8 11F110                      LD      DE,NAME
E9BB DF                          RST     3
E9BC                     ;
E9BC                     ; LOAD & EXEC ADDR CHECK
E9BC                     ;
E9BC AF                          XOR     A
E9BD 32A311                      LD      (BUFER),A
E9C0 2A0411                      LD      HL,(DTADR)
```

```
** Z80 ASSEMBLER 2Z-004C <9Z-502M> PAGE 08                    84.06.26

E9C3 7C                 LD      A,H
E9C4 B5                 OR      L
E9C5 2012               JR      NZ,RDTB0
E9C7 2A0611             LD      HL,(EXADR)
E9CA 7C                 LD      A,H
E9CB B5                 OR      L
E9CC 200B               JR      NZ,RDTB0
E9CE 3EFF               LD      A,FFH
E9D0 32A311             LD      (BUFER),A
E9D3            ;
E9D3            ; LOAD ADRS SET
E9D3            ;
E9D3 210012             LD      HL,1200H
E9D6 220411             LD      (DTADR),HL
E9D9            ;
E9D9            ; READ DATA BLOCK
E9D9            ;
E9D9 CD2A00     RDTB0:  CALL    RDDAT
E9DC DA01EA             JP      C,TPERX
E9DF            ;
E9DF            ; EXEC LOAD FILE
E9DF            ;
E9DF 3AA311     EXLDFL: LD      A,(BUFER)
E9E2 FEFF               CP      FFH
E9E4 2807               JR      Z,EXCLDF
E9E6 010001             LD      BC,0100H          ;DEFAULT(CMT) SET
E9E9 2A0611             LD      HL,(EXADR)
E9EC E9                 JP      (HL)
E9ED            ;
E9ED D3E0       EXCLDF: OUT     (E0H),A
E9EF 210012             LD      HL,1200H
E9F2 110000             LD      DE,0000H
E9F5 ED4B0211           LD      BC,(SIZE)
E9F9 EDB0               LDIR
E9FB 010001             LD      BC,0100H          ;DEFAULT(CMT) SET
E9FE C30000             JP      0000H
EA01            ;
EA01            ; TAPE ERROR
EA01            ;
EA01 FE02       TPERX:  CP      02H               ;BREAK?
EA03 11E6ED     TPERX0: LD      DE,IPLM1          ;'Make ready CMT'
EA06 2803               JR      Z,TPERJP
EA08 115EEE             LD      DE,ERRM2          ;'CMT:Loading error'
EA0B C37BEA     TPERJP: JP      ERRMG
EA0E            ;
EA0E            ;===========================
EA0E            ;
EA0E            ;       QUICK DISK BOOT-UP
EA0E            ;
EA0E            ;===========================
EA0E            ;
EA0E            QBT:    ENT
EA0E CD7AEF             CALL    IOFRS             ;IOCS FLAG RESET
EA11 CDB5EF             CALL    QDRCK             ;MEDIA READY CHECK
EA14 11F7ED             LD      DE,IPLM2          ;'Make ready QD'
EA17 3862               JR      C,ERRMG
EA19 CDA1EA             CALL    DISCLR
EA1C 3E0D               LD      A,0DH
EA1E 32A311             LD      (BUFER),A
EA21 CD18F3             CALL    HDPCL             ;HD POINT CLEAR
```

IPLのリスト

```
EA24                    ;
EA24                    ; ERROR RETURN SET
EA24                    ;
EA24 3E01                       LD      A,01H
EA26 323A11                     LD      (QDCPA),A
EA29 2154EA                     LD      HL,QDLDER
EA2C 31EE10                     LD      SP,10EEH
EA2F E3                         EX      (SP),HL
EA30                    ;
EA30                    ;
EA30 CD85EF             QBFLSH: CALL    FILSCH          ;FILE SEARCH
EA33 DABBF2                     JP      C,QERTRT
EA36 3AF010                     LD      A,(ATRB)
EA39 FE01                       CP      OBJCD
EA3B 1185EE                     LD      DE,ERRM4        ;'QD:File mode error'
EA3E 200A                       JR      NZ,QDFMER
EA40                    ;
EA40                    ; QUICK BOOT
EA40                    ;
EA40 11D4ED                     LD      DE,IPLMO        ;'Ipl is loading'
EA43 DF                         RST     3
EA44 C324EF                     JP      DSFLNA
EA47 11F7ED             QDNTR:  LD      DE,IPLM2        ;'Make ready QD'
EA4A D5                 QDFMER: PUSH    DE
EA4B 3E06                       LD      A,06H           ;MOTOR OFF
EA4D 323011                     LD      (QDPA),A
EA50 CD0DE8                     CALL    QDIOS
EA53 D1                         POP     DE
EA54 1825               QDLDER: JR      ERRMG
EA56                    ;
EA56                    ;======================
EA56                    ;
EA56                    ;       DISP MESSAGE
EA56                    ;
EA56                    ;======================
EA56                    ;
EA56                    ;
EA56                    ; SELECTION MESSAGE
EA56                    ;
EA56 CD0600             SELMSG: CALL    LETNL
EA59 1185ED                     LD      DE,SELMO        ;'Please push key'
EA5C CD95EA                     CALL    MSGLET
EA5F CD0600                     CALL    LETNL
EA62 1197ED                     LD      DE,SELM1        ;'F:Floppy disk'
EA65 CD95EA                     CALL    MSGLET
EA68 11A7ED                     LD      DE,SELM2        ;'Q:Quick disk'
EA6B CD95EA                     CALL    MSGLET
EA6E 11B6ED                     LD      DE,SELM3        ;'C:Cassette tape'
EA71 CD95EA                     CALL    MSGLET
EA74 11C8ED                     LD      DE,SELM4        ;'M:Rom monitor'
EA77 CD95EA                     CALL    MSGLET
EA7A C9                         RET
EA7B                    ;
EA7B                    ; ERROR MESSAGE
EA7B                    ;
EA7B CDA1EA             ERRMG:  CALL    DISCLR
EA7E CD0600                     CALL    LETNL
EA81 CD0600                     CALL    LETNL
EA84 CD95EA                     CALL    MSGLET
EA87 C342E9                     JP      SELMGO
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 10            84.06.26

EA8A                   ;
EA8A                   ; DISP CLEAR 3-LET
EA8A                   ;
EA8A CDA1EA            CLLET:  CALL    DISCLR
EA8D 0603                      LD      B,03H
EA8F CD0600            LET:    CALL    LETNL
EA92 10FB                      DJNZ    LET
EA94 C9                        RET
EA95                   ;
EA95                   ; LETNL-MESSAGE
EA95                   ;
EA95 060C              MSGLET: LD      B,0CH
EA97 CD0C00            MSGPS:  CALL    PRNTS
EA9A 10FB                      DJNZ    MSGPS
EA9C DF                        RST     3
EA9D CD0600                    CALL    LETNL
EAA0 C9                        RET
EAA1                   ;
EAA1                   ; DISP CLEAR
EAA1                   ;
EAA1 3EC6              DISCLR: LD      A,C6H
EAA3 CDDC0D                    CALL    ?DPCT
EAA6 C9                        RET
EAA7                   ;
EAA7                   ;==========================
EAA7                   ;
EAA7                   ;     MONITOR COMMAND
EAA7                   ;
EAA7                   ;==========================
EAA7                   ;
EAA7                   ;
EAA7                   ;  GET COMMAND ROUTINE
EAA7                   ;
EAA7                   ST1X:   ENT
EAA7 CD0900                    CALL    NL
EAAA 3E2A                      LD      A,2AH
EAAC CD1200                    CALL    PRNT
EAAF 11A311                    LD      DE,BUFER
EAB2 CD0300                    CALL    GETL
EAB5 1A                ST2X:   LD      A,(DE)
EAB6 13                        INC     DE
EAB7 FE0D                      CP      0DH
EAB9 CAA7EA                    JP      Z,ST1X
EABC FE4A                      CP      'J'             ;JUMP
EABE CA4DEC                    JP      Z,GOTOX
EAC1 FE4C                      CP      'L'             ;LOAD CMT
EAC3 CA74EB                    JP      Z,LOADX
EAC6 FE46                      CP      'F'             ;FLOPPY BOOT
EAC8 286F                      JR      Z,FDCK
EACA FE42                      CP      'B'             ;BELL
EACC CAA5EC                    JP      Z,SGX
EACF FE23                      CP      '#'             ;CHANGE MEMORY
EAD1 CAA2EC                    JP      Z,CMYOX
EAD4 FE50                      CP      'P'             ;PRINTER
EAD6 CAA9EB                    JP      Z,PTESTX
EAD9 FE4D                      CP      'M'             ;MEMORY CORRECTION
EADB CA1AEC                    JP      Z,MCORX
EADE FE53                      CP      'S'             ;SAVE CMT
EAE0 CA51EC                    JP      Z,SAVEX
EAE3 FE56                      CP      'V'             ;VERIFY
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 11                    84.06.26

EAE5 CA95EC              JP      Z,VRFYX
EAE8 FE44                CP      'D'              ;DUMP MEMORY
EAEA CAB1EC              JP      Z,DUMPX
EAED FE51                CP      'Q'              ;QUICK DISK CMD.
EAEF 2806                JR      Z,QUICK
EAF1 FE45                CP      'E'              ;EXIT RAMBRD(1R12) CMD.
EAF3 2829                JR      Z,RAMBRD
EAF5 18BE                JR      ST2X
EAF7 210000     QUICK:   LD      HL,0000H
EAFA 223A11              LD      (QDCPA),HL       ;FLAG RESET
EAFD 1A                  LD      A,(DE)
EAFE FE4C                CP      'L'              ;LOAD QD
EB00 CA06EF              JP      Z,QL
EB03 FE53                CP      'S'              ;SAVE QD
EB05 CAC1EF              JP      Z,QS
EB08 FE43                CP      'C'              ;COPY QD
EB0A CAB8F1              JP      Z,QC
EB0D FE46                CP      'F'              ;FORMAT QD
EB0F CA3EF1              JP      Z,QF
EB12 FE58                CP      'X'              ;XFER (CMT→QD)
EB14 CA31F2              JP      Z,QX
EB17 FE44                CP      'D'              ;DIRECTRY
EB19 CA87F0              JP      Z,QD
EB1C 1889       JST1X:   JR      ST1X
EB1E 0EF8       RAMBRD:  LD      C,F8H
EB20 CD2EF8              CALL    R12CK1
EB23 3007                JR      NC,RMBD1
EB25 0EA8                LD      C,A8H
EB27 CD2EF8              CALL    R12CK1
EB2A 38F0                JR      C,JST1X
EB2C 1A         RMBD1:   LD      A,(DE)
EB2D FE42                CP      'B'
EB2F CA2DF7              JP      Z,1R12RD         ;1R12 BOOT-UP
EB32 FE53                CP      'S'
EB34 CABEF7              JP      Z,1R12WR         ;1R12 WRITE
EB37 18E3                JR      JST1X
EB39 1A         FDCK:    LD      A,(DE)
EB3A FE0D                CP      0DH
EB3C 20DE                JR      NZ,JST1X
EB3E CD18E9              CALL    FDCC&            ;FD I/O CHECK
EB41 20D9                JR      NZ,JST1X
EB43 CD7FF4              CALL    FDX
EB46 CD0900              CALL    NL
EB49 DF                  RST     3
EB4A 18D0                JR      JST1X
EB4C            ;
EB4C            ;  ERROR (LOADING)
EB4C            ;
EB4C            ?ERX:    ENT
EB4C 1133F4              LD      DE,MGBRK         ;'Break!'
EB4F FE02                CP      02H
EB51 2803                JR      Z,?ERX0
EB53 114701              LD      DE,MSGE1         ;'Check sum err.'
EB56 CD0900     ?ERX0:   CALL    NL
EB59 DF                  RST     3
EB5A 18C0                JR      JST1X
EB5C            ;
EB5C            ; GETLINE & BREAK IN CHECK
EB5C            ;      EXIT BREAK IN THEN JUMP (ST1X)
EB5C            ;      ACC=TOP OF LINE DATA
```

```
**  Z80 ASSEMBLER 2Z-004C  <97-502M>  PAGE 12                84.06.26

EB5C                     ;
EB5C                     BGETLX: ENT
EB5C E3                          EX     (SP),HL
EB5D C1                          POP    BC
EB5E 11A311                      LD     DE,BUFER
EB61 CD0300                      CALL   GETL
EB64 1A                          LD     A,(DE)
EB65 FE1B                        CP     1BH
EB67 28B3                        JR     Z,JST1X
EB69 E9                          JP     (HL)
EB6A                     ;
EB6A                     ; ASCII TO HEX CONVERT
EB6A                     ;      INPUT (DE)=ASCII
EB6A                     ;      CY=1 THEN JUMP (ST1X)
EB6A                     ;
EB6A                     HEXIYX: ENT
EB6A FDE3                        EX     (SP),IY
EB6C F1                          POP    AF
EB6D CD1004                      CALL   HLHEX
EB70 38AA                        JR     C,JST1X
EB72 FDE9                        JP     (IY)
EB74                     ;
EB74                     ;  LOAD COMMAND
EB74                     ;
EB74                     LOADX:  ENT
EB74                     ;
EB74                     ; READ INFORMATION BLOCK
EB74                     ;
EB74 CDD804                      CALL   ?RDI
EB77 38D3                        JR     C,?ERX
EB79                     ;
EB79                     ; DISP 'Loading...'
EB79                     ;
EB79 CD0900              LOAOX:  CALL   NL
EB7C 11FBEE                      LD     DE,MSGLD          ;'loading '
EB7F DF                          RST    3
EB80 11F110                      LD     DE,NAME
EB83 DF                          RST    3
EB84                     ;
EB84                     ; LOAD & EXEC ADRS CHECK
EB84                     ;
EB84 AF                          XOR    A
EB85 32A311                      LD     (BUFER),A
EB88 2A0411                      LD     HL,(DTADR)
EB8B 7C                          LD     A,H
EB8C B5                          OR     L
EB8D 2012                        JR     NZ,RDDTBL
EB8F 2A0611                      LD     HL,(EXADR)
EB92 7C                          LD     A,H
EB93 B5                          OR     L
EB94 200B                        JR     NZ,RDDTBL
EB96 3EFF                        LD     A,FFH
EB98 32A311                      LD     (BUFER),A
EB9B                     ;
EB9B                     ; LOAD ADRS SET
EB9B                     ;
EB9B 210012                      LD     HL,1200H
EB9E 220411                      LD     (DTADR),HL
EBA1                     ;
EBA1                     ; READ DATA BLOCK
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 13                    84.06.26

EBA1                     ;
EBA1 CDF804              RDDTBL: CALL    ?RDD
EBA4 38A6                        JR      C,?ERX
EBA6                     ;
EBA6                     ; EXEC LOAD FILE
EBA6                     ;
EBA6 C3DFE9                      JP      EXLDFL
EBA9                     ;
EBA9                     ;  PRINTER TEST COMMAND
EBA9                     ;
EBA9                     PTESTX: ENT
EBA9 1A                          LD      A,(DE)
EBAA FE26                        CP      '&'
EBAC 2016                        JR      NZ.PTST1X
EBAE 13                  PTSTOX: INC     DE
EBAF 1A                          LD      A,(DE)
EBB0 FE4C                        CP      'L'
EBB2 2816                        JR      Z,.LPTX
EBB4 FE53                        CP      'S'
EBB6 2817                        JR      Z,..LPTX
EBB8 FE43                        CP      'C'
EBBA 2823                        JR      Z,PENX
EBBC FE47                        CP      'G'
EBBE 2818                        JR      Z,PLOTX
EBC0 FE54                        CP      'T'
EBC2 2810                        JR      Z,PTRNX
EBC4 CDF9EB              PTST1X: CALL    PMSG
EBC7 C3A7EA              PTCJST: JP      ST1X
EBCA 117004              .LPTX:  LD      DE,LLPT
EBCD 18F5                        JR      PTST1X
EBCF 11D503              ..LPTX: LD      DE,SLPT
EBD2 18F0                        JR      PTST1X
EBD4 3E04                PTRNX:  LD      A,04H
EBD6 1802                        JR      PLOTX+2
EBD8 3E02                PLOTX:  LD      A,02H
EBDA CDE3EB                      CALL    LPRNT
EBDD 18CF                        JR      PTSTOX
EBDF 3E1D                PENX:   LD      A,1DH
EBE1 18F7                        JR      PLOTX+2
EBE3                     ;
EBE3                     ;
EBE3                     ;       1CHA. PRINT TO $LPT
EBE3                     ;
EBE3                     ;         IN: ACC PRINT DATA
EBE3                     ;
EBE3                     ;
EBE3 0E00                LPRNT:  LD      C,00H
EBE5 47                          LD      B,A
EBE6 CD0AEC                      CALL    RDA
EBE9 78                          LD      A,B
EBEA D3FF                        OUT     (FFH),A          ;DATA OUT
EBEC 3E80                        LD      A,80H            ;RDP HIGH
EBEE D3FE                        OUT     (FEH),A
EBF0 0E01                        LD      C,01H            ;RDA TEST
EBF2 CD0AEC                      CALL    RDA
EBF5 AF                          XOR     A                ;RDP LOW
EBF6 D3FE                        OUT     (FEH),A
EBF8 C9                          RET
EBF9                     ;
EBF9                     ;     $LPT   MSG
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 14          84.06.26

EBF9                    ;      IN: DE DATA LOW ADR.
EBF9                    ;        0DH  MSG.  END
EBF9                    ;
EBF9 D5                 PMSG:   PUSH    DE
EBFA C5                         PUSH    BC
EBFB F5                         PUSH    AF
EBFC 1A                 PMSG1:  LD      A,(DE)          ;ACC=DATA
EBFD CDE3EB                     CALL    LPRNT
EC00 1A                         LD      A,(DE)
EC01 13                         INC     DE
EC02 FE0D                       CP      0DH
EC04 20F6                       JR      NZ,PMSG1
EC06 F1                         POP     AF
EC07 C1                         POP     BC
EC08 D1                         POP     DE
EC09 C9                         RET
EC0A                    ;
EC0A                    ;    RDA CHECK
EC0A                    ;
EC0A                    ;        BREAK IN TO MONITOR RETURN
EC0A                    ;        IN: C RDA CODE
EC0A                    ;
EC0A DBFE               RDA:    IN      A,(FEH)
EC0C E60D                       AND     0DH
EC0E B9                         CP      C
EC0F C8                         RET     Z
EC10 CD1E00                     CALL    BRKEY
EC13 20F5                       JR      NZ,RDA
EC15 31F010                     LD      SP,SP
EC18 18AD                       JR      PTCJST
EC1A                    ;
EC1A                    ;   MEMORY CORRECTION
EC1A                    ;
EC1A                    MCORX:  ENT
EC1A CD6AEB                     CALL    HEXIYX
EC1D                    MCR1X:  ENT
EC1D CDFA05                     CALL    NLPHL
EC20 CDB103                     CALL    SPHEX
EC23 CD2009                     CALL    ?PRTS
EC26 CD5CEB                     CALL    BGETLX
EC29 CD1004                     CALL    HLHEX
EC2C 381B                       JR      C,MCR3X
EC2E CDA602                     CALL    .4DE
EC31 13                         INC     DE
EC32 CD1F04                     CALL    2HEX
EC35 38E6                       JR      C,MCR1X
EC37 BE                         CP      (HL)
EC38 20E3                       JR      NZ,MCR1X
EC3A 13                         INC     DE
EC3B 1A                         LD      A,(DE)
EC3C FE0D                       CP      0DH
EC3E 2806                       JR      Z,MCR2X
EC40 CD1F04                     CALL    2HEX
EC43 38D8                       JR      C,MCR1X
EC45 77                         LD      (HL),A
EC46                    MCR2X:  ENT
EC46 23                         INC     HL
EC47 18D4                       JR      MCR1X
EC49 60                 MCR3X:  LD      H,B
EC4A 69                         LD      L,C
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 15                84.06.26

EC4B 18D0                    JR     MCR1X
EC4D                 ;
EC4D                 ;  JUMP COMMAND
EC4D                 ;
EC4D CD6AEB          GOTOX:  CALL   HEXIYX
EC50 E9                      JP     (HL)
EC51                 ;
EC51                 ;  SAVE COMMAND
EC51                 ;
EC51 CD16ED          SAVEX:  CALL   FNINP
EC54 21A311                  LD     HL,BUFER
EC57 11F110                  LD     DE,NAME
EC5A 011100                  LD     BC,0011H
EC5D EDB0                    LDIR
EC5F CD5CED                  CALL   TAINP
EC62 220411                  LD     (DTADR),HL      ;DATA ADDR
EC65 CD61ED                  CALL   EAINP
EC68 ED4B0411                LD     BC,(DTADR)
EC6C 37                      SCF
EC6D ED42                    SBC    HL,BC
EC6F 23                      INC    HL
EC70 23                      INC    HL
EC71 220211                  LD     (SIZE),HL       ;BYTE SIZE
EC74 CD66ED                  CALL   XAINP
EC77 220611                  LD     (EXADR),HL      ;EXEC ADDR
EC7A 3E01                    LD     A,01H
EC7C 32F010                  LD     (ATRB),A
EC7F CD3604                  CALL   ?WRI
EC82 DA4CEB                  JP     C,?ERX
EC85 CD7504                  CALL   ?WRD
EC88 DA4CEB                  JP     C,?ERX
EC8B CD0900                  CALL   NL
EC8E 114209                  LD     DE,MSGOK
EC91 DF                      RST    3
EC92 C3A7EA                  JP     ST1X
EC95                 ;
EC95                 ;  VERIFYING COMMAND
EC95                 ;
EC95                 VRFYX:  ENT
EC95 CD8805                  CALL   ?VRFY
EC98 DA4CEB                  JP     C,?ERX
EC9B 114209                  LD     DE,MSGOK
EC9E DF                      RST    3
EC9F C3A7EA                  JP     ST1X
ECA2                 ;
ECA2                 ;  CHANG MEMORY COMMAND
ECA2                 ;
ECA2                 CMYOX:  ENT
ECA2 C35B00                  JP     CMYO
ECA5                 ;
ECA5                 ;  KEYIN BELL COMMAND 'B'
ECA5                 ;
ECA5 3A9D11          SGX:    LD     A,(SWRK)
ECA8 1F                      RRA
ECA9 3F                      CCF
ECAA 17                      RLA
ECAB 329D11                  LD     (SWRK),A
ECAE C3A7EA                  JP     ST1X
ECB1                 ;
ECB1                 ;  MEMORY DUMP COMMAND
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 16          84.06.26

ECB1                 ;
ECB1                 DUMPX:  ENT
ECB1 CD6AEB                  CALL    HEXIYX
ECB4 CDA602                  CALL    .4DE
ECB7 E5                      PUSH    HL
ECB8 CD1004                  CALL    HLHEX
ECBB D1                      POP     DE
ECBC 3852                    JR      C,DUM1X
ECBE EB                      EX      DE,HL
ECBF 0608            DUM3X:  LD      B,08H
ECC1 0E17                    LD      C,17H
ECC3 CDFA05                  CALL    NLPHL
ECC6 CDB103          DUM2X:  CALL    SPHEX
ECC9 23                      INC     HL
ECCA F5                      PUSH    AF
ECCB 3A7111                  LD      A,(DSPXY)
ECCE 81                      ADD     A,C
ECCF 327111                  LD      (DSPXY),A
ECD2 F1                      POP     AF
ECD3 FE20                    CP      20H
ECD5 3002                    JR      NC,+4
ECD7 3E2E                    LD      A,2EH
ECD9 CDB90B                  CALL    ?ADCN
ECDC CD6C09                  CALL    PRNT3
ECDF 3A7111                  LD      A,(DSPXY)
ECE2 0C                      INC     C
ECE3 91                      SUB     C
ECE4 327111                  LD      (DSPXY),A
ECE7 0D                      DEC     C
ECE8 0D                      DEC     C
ECE9 0D                      DEC     C
ECEA E5                      PUSH    HL
ECEB ED52                    SBC     HL,DE
ECED E1                      POP     HL
ECEE 281D                    JR      Z,DUM1X-3
ECF0 3EF8                    LD      A,F8H
ECF2 3200E0                  LD      (KEYPA),A
ECF5 00                      NOP
ECF6 3A01E0                  LD      A,(KEYPB)
ECF9 FEFE                    CP      FEH
ECFB 2003                    JR      NZ,+5
ECFD CDA60D                  CALL    ?BLNK
ED00 10C4                    DJNZ    DUM2X
ED02 CDCA08                  CALL    ?KEY
ED05 B7                      OR      A
ED06 28FA                    JR      Z,-4
ED08 CD320A                  CALL    ?BRK
ED0B 20B2                    JR      NZ,DUM3X
ED0D C3A7EA                  JP      ST1X
ED10 21A000          DUM1X:  LD      HL,00A0H
ED13 19                      ADD     HL,DE
ED14 18A8                    JR      DUM3X-1
ED16                 ;
ED16                 ; INPUT FILE NAME
ED16                 ;
ED16                 FNINP:  ENT
ED16 CD0900                  CALL    NL
ED19 11C7EE                  LD      DE,MSGSV0          ;'Filename?'
ED1C DF                      RST     3
ED1D 11A311                  LD      DE,BUFER
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 17                    84.06.26

ED20 CD0300                     CALL    GETL
ED23 1A                         LD      A,(DE)
ED24 FE1B                       CP      1BH             ;BREAK ?
ED26 2005                       JR      NZ,FNINP0
ED28 21A7EA             FNINPR: LD      HL,ST1X
ED2B E3                         EX      (SP),HL
ED2C C9                         RET
ED2D                    ;
ED2D                    ;
ED2D 0600               FNINP0: LD      B,00H
ED2F 11AF11                     LD      DE,BUFER+12
ED32 21A311                     LD      HL,BUFER
ED35 1A                         LD      A,(DE)
ED36 FE0D                       CP      0DH
ED38 2820                       JR      Z,FNINP6
ED3A                    ;
ED3A                    ; SPACE CHECK
ED3A                    ;
ED3A FE20               FNINP1: CP      ' '             ; SPACE ?
ED3C 2004                       JR      NZ,FNINP2
ED3E 13                         INC     DE
ED3F 1A                         LD      A,(DE)
ED40 18F8                       JR      FNINP1
ED42                    ;
ED42                    ;  " ?
ED42                    ;
ED42 FE22               FNINP2: CP      22H
ED44 2808                       JR      Z,FNINP4
ED46                    ;
ED46                    ; TRANS (DE)→(HL)
ED46                    ;
ED46 77                 FNINP3: LD      (HL),A
ED47 23                         INC     HL
ED48 04                         INC     B
ED49 3E11                       LD      A,11H
ED4B B8                         CP      B
ED4C 28C8                       JR      Z,FNINP
ED4E                    ;
ED4E                    ; FILENAME END CHECK
ED4E                    ;
ED4E 13                 FNINP4: INC     DE
ED4F 1A                         LD      A,(DE)
ED50 FE22                       CP      22H             ; " ?
ED52 2804                       JR      Z,FNINP5
ED54 FE0D                       CP      0DH
ED56 20EE                       JR      NZ,FNINP3
ED58 3E0D               FNINP5: LD      A,0DH
ED5A 77                 FNINP6: LD      (HL),A
ED5B C9                         RET
ED5C                    ;
ED5C                    ; INPUT SAVE CONDITION
ED5C                    ;
ED5C                    TAINP:  ENT
ED5C 11D4EE                     LD      DE,MSGTA        ;'TOP ADRS?'
ED5F 1808                       JR      AINP0
ED61                    EAINP:  ENT
ED61 11E1EE                     LD      DE,MSGEA        ;'END ADRS?'
ED64 1803                       JR      AINP0
ED66                    XAINP:  ENT
ED66 11EEEE                     LD      DE,MSGXA        ;'EXC ADRS?'
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 18           84.06.26

ED69 CD0900         AINPO:  CALL   NL
ED6C DF                     RST    3
ED6D D5                     PUSH   DE
ED6E 11A311                 LD     DE,BUFER
ED71 CD0300                 CALL   GETL
ED74 1A                     LD     A,(DE)
ED75 FE1B                   CP     1BH              ;BREAK ?
ED77 D1                     POP    DE
ED78 28AE                   JR     Z,FNINPR
ED7A D5                     PUSH   DE
ED7B 11AF11                 LD     DE,BUFER+12
ED7E CD1004                 CALL   HLHEX
ED81 D1                     POP    DE
ED82 38E5                   JR     C,AINPO
ED84 C9                     RET
ED85                ;
ED85                ;
ED85 50054C45       SELM0:  DEFM   'P□LEASE PUSH KEY□'
ED89 41534520
ED8D 50555348
ED91 204B4559
ED95 05
ED96 0D                     DEFB   0DH
ED97 463A4605       SELM1:  DEFM   'F:F□LOPPY DISK□'
ED9B 4C4F5050
ED9F 59204449
EDA3 534B05
EDA6 0D                     DEFB   0DH
EDA7 513A5105       SELM2:  DEFM   'Q:Q□UICK DISK□'
EDAB 5549434B
EDAF 20444953
EDB3 4B05
EDB5 0D                     DEFB   0DH
EDB6 433A4305       SELM3:  DEFM   'C:C□ASSETTE TAPE□'
EDBA 41535345
EDBE 54544520
EDC2 54415045
EDC6 05
EDC7 0D                     DEFB   0DH
EDC8 4D3A4D05       SELM4:  DEFM   'M:M□ONITOR□'
EDCC 4F4E4954
EDD0 4F5205
EDD3 0D                     DEFB   0DH
EDD4                IPLM0:  ENT
EDD4 49504C20               DEFM   'IPL □IS LOADING□ '
EDD8 05495320
EDDC 4C4F4144
EDE0 494E4705
EDE4 20
EDE5 0D                     DEFB   0DH
EDE6 4D05414B       IPLM1:  DEFM   'M□AKE READY□ CMT'
EDEA 45205245
EDEE 41445905
EDF2 20434D54
EDF6 0D                     DEFB   0DH
EDF7 4D05414B       IPLM2:  DEFM   'M□AKE READY□ QD'
EDFB 45205245
EDFF 41445905
EE03 205144
EE06 0D                     DEFB   0DH
```

IPLのリスト

```
EE07                    IPLM3:  ENT
EE07 4D05414B                   DEFM   'M□AKE READY□ FD'
EE0B 45205245
EE0F 41445905
EE13 204644
EE16 0D                         DEFB   0DH
EE17 20202020           IPLM4:  DEFM   '    IPL □IS LOOKING FOR A PROGRAM□'
EE1B 2049504C
EE1F 20054953
EE23 204C4F4F
EE27 4B494E47
EE2B 20464F52
EE2F 20412050
EE33 524F4752
EE37 414D05
EE3A 0D                         DEFB   0DH
EE3B                    ERRM0:  ENT
EE3B 46443A4C                   DEFM   'FD:L□OADING ERROR□'
EE3F 054F4144
EE43 494E4720
EE47 4552524F
EE4B 5205
EE4D 0D                         DEFB   0DH
EE4E                    ERRM1:  ENT
EE4E 46443A4E                   DEFM   'FD:N□OT MASTER□'
EE52 054F5420
EE56 4D415354
EE5A 455205
EE5D 0D                         DEFB   0DH
EE5E 434D543A           ERRM2:  DEFM   'CMT:L□OADING ERROR□'
EE62 4C054F41
EE66 44494E47
EE6A 20455252
EE6E 4F5205
EE71 0D                         DEFB   0DH
EE72 51443A4C           ERRM3:  DEFM   'QD:L□OADING ERROR□'
EE76 054F4144
EE7A 494E4720
EE7E 4552524F
EE82 5205
EE84 0D                         DEFB   0DH
EE85 51443A46           ERRM4:  DEFM   'QD:F□ILE MODE ERROR□'
EE89 05494C45
EE8D 204D4F44
EE91 45204552
EE95 524F5205
EE99 0D                         DEFB   0DH
EE9A 5352414D           12ERMG: DEFM   'SRAM:C□HECK SUM ERR□'
EE9E 3A430548
EEA2 45434B20
EEA6 53554D20
EEAA 45525205
EEAE 0D                         DEFB   0DH
EEAF 2A2A2020           MONMSG: DEFM   '**  MONITOR 9Z-502M   **'
EEB3 4D4F4E49
EEB7 544F5220
EEBB 395A2D35
EEBF 30324D20
EEC3 202A2A
EEC6 0D                         DEFB   0DH
```

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 20              84.06.26

EEC7 4605494C            MSGSVO: DEFM    ´F□ILENAME□? ´
EECB 454E414D
EECF 45053F20
EED3 OD                          DEFB    ODH
EED4 54054F50            MSGTA:  DEFM    ´T□OP ADRS□? ´
EED8 20414452
EEDC 53053F20
EEE0 OD                          DEFB    ODH
EEE1 45054E44            MSGEA:  DEFM    ´E□ND ADRS□? ´
EEE5 20414452
EEE9 53053F20
EEED OD                          DEFB    ODH
EEEE 45055843            MSGXA:  DEFM    ´E□XC ADRS□? ´
EEF2 20414452
EEF6 53053F20
EEFA OD                          DEFB    ODH
EEFB                     MSGLD:  ENT
EEFB 4C054F41                    DEFM    ´L□OADING□ ´
EEFF 44494E47
EF03 0520
EF05 OD                          DEFB    ODH
EF06                     ;
EF06                     ;
EF06                             SKP     H
```

IPLのリスト

```
** Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 21                84.06.26

EF06                      ;
EF06                      ;
EF06                      ;
EF06                      ; QUICK DISK CONTROL COMMAND
EF06                      ;           QL=QUICK LOAD
EF06                      ;           QS=QUICK SAVE
EF06                      ;           QD=QUICK DIRECTORY
EF06                      ;           QF=QUICK FORMATTING
EF06                      ;           QX=QUICK XFER (CMT→DISK)
EF06                      ;           QC=QUICK COPY (DISK→DISK)
EF06                      ;
EF06                      ;
11A3   P                  BUFF:   EQU    11A3H          ;KEYIN BUFFER
EF06                      ;
EF06                      ;============================
EF06                      ;
EF06                      ;   QUICK DISK LOAD COMMAND
EF06                      ;
EF06                      ;============================
EF06                      ;
EF06                      QL:     ENT
EF06 CD7AEF                       CALL   IOFRS          ;IOCS FLAG RESET
EF09 CDB5EF                       CALL   QDRCK          ;READY CHECK
EF0C 3847                         JR     C,QER00
EF0E CD16ED                       CALL   FNINP          ;INPUT FILENAME
EF11 CD18F3                       CALL   HDPCL          ;HEAD POINT CLEAR
EF14                      ;
EF14                      ; DISP 'Loading...'
EF14                      ;
EF14 11FBEE                       LD     DE,MSGLD       ;'Loading'
EF17 DF                           RST    3
EF18                      ;
EF18                      ; FILE SEARCH
EF18                      ;
EF18 CD85EF               FILESH: CALL   FILSCH
EF1B 3838                         JR     C,QER00
EF1D                      ;
EF1D                      ; ATRIBUTE CHECK
EF1D                      ;
EF1D 3AF010               QDATRC: LD     A,(ATRB)
EF20 FE01                         CP     OBJCD
EF22 20F4                         JR     NZ,FILESH
EF24                      ;
EF24                      ; DISP 'LOADING ......'
EF24                      ;
EF24                      DSFLNA: ENT
EF24 11F110                       LD     DE,NAME
EF27 DF                           RST    3
EF28                      ;
EF28                      ; LOAD & EXEC ADDR CHECK
EF28                      ;
EF28 2A0611                       LD     HL,(QDTADR)
EF2B 7C                           LD     A,H
EF2C B5                           OR     L
EF2D 2005                         JR     NZ,LEAC0
EF2F 2A0811                       LD     HL,(QEXADR)
EF32 7C                           LD     A,H
EF33 B5                           OR     L
EF34 200A                 LEAC0:  JR     NZ,LPARA0
EF36 3EFF                         LD     A,FFH
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 22                  84.06.26

EF38 323B11               LD      (QDCPB),A
EF3B                  ;
EF3B                  ; IOCS PARAMETER SET
EF3B                  ;
EF3B 210012               LD      HL,1200H
EF3E 1803                 JR      LPARA1
EF40 2A0611       LPARA0: LD      HL,(QDTADR)
EF43 223211       LPARA1: LD      (QDPC),HL         ;DATA ADDR SET
EF46 2A0411               LD      HL,(QSIZE)
EF49 223411               LD      (QDPE),HL
EF4C 210301               LD      HL,0103H          ;READ DATA BLOCK CMD.
EF4F 223011               LD      (QDPA),HL
EF52                  ;
EF52                  ; READ DATA BLOCK
EF52                  ;
EF52 CD0DE8               CALL    QDIOS             ;QD IOCS
EF55 DA07F0       QER00:  JP      C,QER02
EF58                  ;
EF58                  ; EXEC LOAD FILE
EF58                  ;
EF58 3A3B11               LD      A,(QDCPB)         ;LOAD=EXEC=0000H ?
EF5B FEFF                 CP      FFH
EF5D 2807                 JR      Z,EXLDF
EF5F 010003               LD      BC,0300H          ;DEFAULT(QD) SET
EF62 2A0811               LD      HL,(QEXADR)
EF65 E9                   JP      (HL)
EF66                  ;
EF66 D3E0         EXLDF:  OUT     (E0H),A
EF68 210012               LD      HL,1200H          ;BLOCK TRANS
EF6B 110000               LD      DE,0000H
EF6E ED4B0411             LD      BC,(QSIZE)
EF72 EDB0                 LDIR
EF74 010003               LD      BC,0300H          ;DEFAULT(QD) SET
EF77 C30000               JP      0000H
EF7A                  ;
EF7A                  ; IOCS FLAG RESET
EF7A                  ;
EF7A                  IOFRS:  ENT
EF7A AF                   XOR     A                 ;Acc←00H
EF7B 324411               LD      (MTF),A           ;MOTOR FLAG
EF7E 323F11               LD      (FNUPS),A         ;FILE NUMBER FLAG
EF81 324111               LD      (FNUPF),A         ;FILE NUMBER UP FLAG
EF84 C9                   RET
EF85                  ;
EF85                  ;
EF85                  ; FILE SEARCH SUB.
EF85                  ;
EF85                  ;
EF85                  FILSCH: ENT
EF85                  ;
EF85                  ; IOCS PARAMETER SET
EF85                  ;
EF85 210300               LD      HL,0003H          ;READ INFORMATION CMD.
EF88 223011               LD      (QDPA),HL
EF8B 21F010               LD      HL,10F0H          ;HEAD ADDR
EF8E 223211               LD      (QDPC).HL
EF91 214000               LD      HL.0040H          ;READ SIZE
EF94 223411               LD      (QDPE),HL
EF97                  ;
EF97                  ; READ INFORMATION BLOCK
```

```
**  Z80 ASSEMBLER ZZ-004C  <9Z-502M>  PAGE 23                84.06.26

EF97                  ;
EF97 CD0DE8           QLINF:  CALL   QDIOS
EF9A D8                       RET    C
EF9B                  ;
EF9B                  ; FILE NAME CHECK
EF9B                  ;
EF9B 3AA311                   LD     A,(BUFF)
EF9E FE0D                     CP     0DH
EFA0 C8                       RET    Z                 ;CY=0
EFA1 21A311                   LD     HL,BUFF
EFA4 11F110                   LD     DE,NAME
EFA7 0611                     LD     B,NAMSIZ
EFA9 1A               LDFNCK: LD     A,(DE)
EFAA BE                       CP     (HL)
EFAB 20EA                     JR     NZ,QLINF
EFAD FE0D                     CP     0DH
EFAF C8                       RET    Z                 ;CY=0
EFB0 13                       INC    DE
EFB1 23                       INC    HL
EFB2 10F5                     DJNZ   LDFNCK
EFB4 C9                       RET
EFB5                  ;
EFB5                  ; QUICK DISK READY CHECK
EFB5                  ;
EFB5                  QDRCK:  ENT
EFB5 AF                       XOR    A
EFB6 323111                   LD     (QDPB),A
EFB9 3C                       INC    A
EFBA 323011                   LD     (QDPA),A
EFBD CD0DE8                   CALL   QDIOS
EFC0 C9                       RET
EFC1                  ;
EFC1                  ;============================
EFC1                  ;
EFC1                  ;   QUICK DISK SAVE COMMAND
EFC1                  ;
EFC1                  ;============================
EFC1                  ;
EFC1                  QS:     ENT
EFC1 CD7AEF                   CALL   IOFRS             ;IOCS FLAG RESET
EFC4 CD79F0                   CALL   QDRWCK            ;READY & WPRT CHECK
EFC7 383E                     JR     C,QER02
EFC9                  ;
EFC9                  ; INPUT FILE NAME
EFC9                  ;
EFC9 CD16ED                   CALL   FNINP             ;INPUT FILE NAME
EFCC 3AA311                   LD     A,(BUFF)
EFCF FE0D                     CP     0DH
EFD1 28EE                     JR     Z,QS
EFD3 21A311                   LD     HL,BUFF
EFD6 11F110                   LD     DE,NAME
EFD9 011100                   LD     BC,0011H
EFDC EDB0                     LDIR
EFDE                  ;
EFDE                  ; INPUT TOP & END & EXC ADDRS
EFDE                  ;
EFDE CD5CED                   CALL   TAINP             ;TOP ADDR
EFE1 220611                   LD     (QDTADR),HL
EFE4 CD61ED                   CALL   EAINP             ;END ADDR
EFE7 B7                       OR     A                 ;CARRY RESET
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 24           84.06.26

EFE8 ED4B0611                 LD      BC,(QDTADR)
EFEC ED42                     SBC     HL,BC
EFEE 23                       INC     HL
EFEF 220411                   LD      (QSIZE),HL
EFF2 CD66ED                   CALL    XAINP             ;EXC ADDR
EFF5 220811                   LD      (QEXADR),HL
EFF8                 ;
EFF8                 ; ATRIBUTE SET
EFF8                 ;
EFF8 3E01                     LD      A,OBJCD           ;OBJ CORD
EFFA 32F010                   LD      (ATRB),A
EFFD                 ;
EFFD                 ; FILE END SEARCH
EFFD                 ;
EFFD CD37F0                   CALL    FEDSCH
F000 3805                     JR      C,QER02
F002                 ;
F002                 ; SAVE FILE START
F002                 ;
F002 FE28            SVFLST:  CP      NTFECD            ;NOT FOUND ?
F004 2803                     JR      Z,QSIOST
F006 37                       SCF
F007 3821            QER02:   JR      C,QER03
F009                 ;
F009                 ; IOCS PARAMETER SET
F009                 ;
F009 2A0611          QSIOST:  LD      HL,(QDTADR)       ;DATA TOP ADDR
F00C 223611          QSVIOS:  LD      (QDPG),HL         ;COPY JUMP POINT
F00F 210404                   LD      HL,0404H          ;WRITE INF+DATA BLK CMD.
F012 223011                   LD      (QDPA),HL
F015 21F010                   LD      HL,ATRB           ;INFM TOP ADDR
F018 223211                   LD      (QDPC),HL
F01B 214000                   LD      HL,0040H          ;INFM BYTE SIZE
F01E 223411                   LD      (QDPE),HL
F021 2A0411                   LD      HL,(QSIZE)        ;DATA BYTE SIZE
F024 223811                   LD      (QDPI),HL
F027                 ;
F027                 ; WRITE INF & DATA BLOCK
F027                 ;
F027 CD0DE8                   CALL    QDIOS
F02A DAC3F0          QER03:   JP      C,QER04
F02D                 ;
F02D                 ; QUICK DISK 'OK' MESSAGE
F02D                 ;
F02D CD0900          QDOKM:   CALL    NL
F030 1121F3                   LD      DE,MSGQOK         ;'OK!'
F033 DF                       RST     3
F034 C3A7EA                   JP      ST1X
F037                 ;
F037                 ; FILE END SEARCH SUB.
F037                 ;
F037                 FEDSCH:  ENT
F037 CD18F3                   CALL    HDPCL             ;HD POINT CLEAR
F03A                 ;
F03A                 ; IOCS PARAMETER SET
F03A                 ;
F03A 210300                   LD      HL,0003H          ;READ INFORMATION CMD.
F03D 223011                   LD      (QDPA),HL
F040 21A311                   LD      HL,BUFF           ;HEAD ADDR
F043 223211                   LD      (QDPC),HL
```

```
** Z80 ASSEMBLER 2Z-004C <9Z-502M> PAGE 25                    84.06.26

F046 214000               LD     HL,0040H          ;READ SIZE
F049 223411               LD     (QDPE),HL
F04C                ;
F04C                ; TOO MANY FILES ?
F04C                ;
F04C AF                   XOR    A
F04D 323C11               LD     (QDCPC),A
F050 3A3C11       EDFLCK: LD     A,(QDCPC)
F053 3C                   INC    A
F054 FE21                 CP     33                ;FILES
F056 323C11               LD     (QDCPC),A
F059 3E33                 LD     A,TMFECD          ;TOO MANY FILES ERR
F05B D0                   RET    NC
F05C                ;
F05C                ; END FILE CHECK
F05C                ;
F05C CD0DE8               CALL   QDIOS
F05F 3F                   CCF                      ;COMP CY
F060 D0                   RET    NC
F061                ;
F061                ; SAME FILE NAME CHECK
F061                ;
F061 11A411               LD     DE,BUFF+1
F064 21F110               LD     HL,NAME
F067 0611                 LD     B,11H
F069 1A           SFNC:   LD     A,(DE)
F06A BE                   CP     (HL)
F06B 20E3                 JR     NZ,EDFLCK
F06D FE0D                 CP     0DH
F06F 2804                 JR     Z,SFNC0
F071 13                   INC    DE
F072 23                   INC    HL
F073 10F4                 DJNZ   SFNC
F075 3E2A         SFNC0:  LD     A,ALEXCD          ;ALREADY EXIST
F077 37                   SCF                      ;CY=1
F078 C9                   RET
F079                ;
F079                ; READY & WRITE PROTECT CHECK
F079                ;
F079 3EFF         QDRWCK: LD     A,FFH
F07B 323111               LD     (QDPB),A
F07E 3E01                 LD     A,01H
F080 323011               LD     (QDPA),A
F083 CD0DE8               CALL   QDIOS
F086 C9                   RET
F087                ;
F087                ;==================================
F087                ;
F087                ;   QUICK DISK DIRECTORY COMMAND
F087                ;
F087                ;==================================
F087                ;
F087              QD:     ENT
F087 CD7AEF               CALL   IOFRS             ;IOCS FLAG RESET
F08A CDB5EF               CALL   QDRCK             ;READY CHECK
F08D 3834                 JR     C,QER04
F08F CD18F3               CALL   HDPCL             ;HD POINT CLEAR
F092 0600                 LD     B,00H             ;COUNTER RESET
F094                ;
F094                ; DISP 'Directory of QD:'
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 26           84.06.26

F094                ;
F094 CD0900                 CALL   NL
F097 116CF4                 LD     DE,DIRMSG
F09A DF                     RST    3
F09B                ;
F09B                ; IOCS PARAMETER SET
F09B                ;
F09B 2190CD                 LD     HL,QDIRBF          ;QD DIR BUFFER
F09E 223211         DIRIOP: LD     (QDPC),HL
F0A1 210300                 LD     HL,0003H           ;READ INFORMATION CMD.
F0A4 223011                 LD     (QDPA),HL
F0A7 214000                 LD     HL,0040H           ;READ SIZE
F0AA 223411                 LD     (QDPE),HL
F0AD                ;
F0AD                ; READ INFOMATION BLOCK
F0AD                ;
F0AD C5                     PUSH   BC
F0AE CD0DE8                 CALL   QDIOS
F0B1 C1                     POP    BC
F0B2 380A                   JR     C,DIREFC
F0B4 04                     INC    B
F0B5                ;
F0B5                ; BUFFER ADDR INCREMENT
F0B5                ;
F0B5 2A3211                 LD     HL,(QDPC)
F0B8 111200                 LD     DE,0012H
F0BB 19                     ADD    HL,DE
F0BC 18E0                   JR     DIRIOP
F0BE                ;
F0BE                ; END FILE CHECK
F0BE                ;
F0BE FE28           DIREFC: CP     NTFECD             ;NOT FOUND ?
F0C0 2804                   JR     Z,DIRMTF
F0C2 37                     SCF
F0C3 DA55F1         QER04:  JP     C,QER05
F0C6                ;
F0C6                ; MOTOR OFF
F0C6                ;
F0C6 3E06           DIRMTF: LD     A,06H              ;MOTOR OFF CMD.
F0C8 323011                 LD     (QDPA),A
F0CB C5                     PUSH   BC
F0CC CD0DE8                 CALL   QDIOS
F0CF C1                     POP    BC
F0D0                ;
F0D0                ; NO FILE CHECK
F0D0                ;
F0D0 AF                     XOR    A
F0D1 B8                     CP     B
F0D2 3067                   JR     NC,QDOKMO
F0D4                ;
F0D4                ; DIRECTORY DISP
F0D4                ;
F0D4 CD0900                 CALL   NL
F0D7 2190CD                 LD     HL,QDIRBF          ;QD DIR BUFFER
F0DA                ;
F0DA                ; DISP ATRIBUTE
F0DA                ;
F0DA 7E             DSPATR: LD     A,(HL)
F0DB 1125F3                 LD     DE,MSGQ01          ;'OBJ'
F0DE 3D                     DEC    A
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 27            84.06.26

F0DF 2836                JR      Z,DRDIS0          ;OBJ FILE
F0E1 1130F3              LD      DE,MSGQ02         ;'BTX'
F0E4 3D                  DEC     A
F0E5 2830                JR      Z,DRDIS0          ;BTX FILE
F0E7 113BF3              LD      DE,MSGQ03         ;'BSD'
F0EA 3D                  DEC     A
F0EB 282A                JR      Z,DRDIS0          ;BSD FILE
F0ED 1146F3              LD      DE,MSGQ04         ;'BRD'
F0F0 3D                  DEC     A
F0F1 2824                JR      Z,DRDIS0          ;BRD FILE
F0F3 1151F3              LD      DE,MSGQ05         ;'RB '
F0F6 3D                  DEC     A
F0F7 281E                JR      Z,DRDIS0          ;RB  FILE
F0F9 3D                  DEC     A
F0FA 2818                JR      Z,DRDIS1          ;??? FILE
F0FC 115CF3              LD      DE,MSGQ07         ;'LIB'
F0FF 3D                  DEC     A
F100 2815                JR      Z,DRDIS0          ;LIB FILE
F102 3D                  DEC     A
F103 280F                JR      Z,DRDIS1          ;??? FILE
F105 3D                  DEC     A
F106 280C                JR      Z,DRDIS1          ;??? FILE
F108 1167F3              LD      DE,MSGQ10         ;'SYS'
F10B 3D                  DEC     A
F10C 2809                JR      Z,DRDIS0          ;SYS FILE
F10E 1172F3              LD      DE,MSGQ11         ;'GR '
F111 3D                  DEC     A
F112 2803                JR      Z,DRDIS0          ;GR  FILE
F114 117DF3      DRDIS1: LD      DE,MSGQ??         ;'???'
F117 DF          DRDIS0: RST     3
F118             ;
F118             ; DISP FILE NAME
F118             ;
F118 3E22                LD      A,'"'
F11A CD1200              CALL    PRNT
F11D 23                  INC     HL
F11E E5                  PUSH    HL
F11F D1                  POP     DE
F120 DF                  RST     3
F121 3E22                LD      A,'"'
F123 CD1200              CALL    PRNT
F126 CD0900              CALL    NL
F129             ;
F129             ; COUNTER DECRIMENT
F129             ;
F129 111100              LD      DE,0011H
F12C 19                  ADD     HL,DE
F12D CDCA08              CALL    ?KEY
F130 B7                  OR      A
F131 28FA                JR      Z,-4
F133 CD320A              CALL    ?BRK
F136 CAA7EA              JP      Z,ST1X
F139 109F                DJNZ    DSPATR            ;B=FILE COUNTER
F13B             ;
F13B             ; DIRECTORY END
F13B             ;
F13B C32DF0      QDOKMO: JP      QDOKM
F13E             ;
F13E             ;
F13E             ;
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 28          84.06.26

F13E                    ;
F13E                    ;==============================
F13E                    ;
F13E                    ;  QUICK DISK FORMAT COMMAND
F13E                    ;
F13E                    ;==============================
F13E                    ;
F13E                    QF:     ENT
F13E                    ;
F13E                    ;
F13E                    ; DISP FORMATTING
F13E                    ;
F13E 113CF4                     LD      DE,QDFMG        ;'QD:FORMATTING'
F141 DF                         RST     3
F142 CD5AF1                     CALL    WAITY           ;'OK?(Y/N)'
F145                    ;
F145                    ; IOCS FLAG RESET
F145                    ;
F145 CD7AEF                     CALL    IOFRS
F148                    ;
F148                    ; READY & WRPT CHECK
F148                    ;
F148 CD79F0                     CALL    QDRWCK
F14B 3808                       JR      C,QERO5
F14D                    ;
F14D                    ; FORMAT
F14D                    ;
F14D 3E02                       LD      A,02H           ;INIT CMD SET
F14F 323011                     LD      (QDPA),A
F152 CD0DE8                     CALL    QDIOS
F155 DAEFF1             QERO5:  JP      C,QERO6
F158                    ;
F158                    ; FORMAT END
F158                    ;
F158 18E1                       JR      QDOKMO
F15A                    ;
F15A                    ; WAIT KEY IN 'Y'
F15A                    ;
F15A CD0900             WAITY:  CALL    NL
F15D 1163F4                     LD      DE,QDQCMO       ;'OK?(Y/N)'
F160 DF                         RST     3
F161 CD20F2                     CALL    NKYWT           ;WAIT NON KEY
F164 CDB10F             WATKY:  CALL    0FB1H           ;CALL ?PONT
F167 3EEF                       LD      A,EFH           ;'▒'
F169 77                 WATKO:  LD      (HL),A
F16A CD77F1                     CALL    WATIM
F16D 3805                       JR      C,WATK1         ;IF 'Y' THEN CY=1
F16F 7E                         LD      A,(HL)
F170 EEEF                       XOR     EFH
F172 18F5                       JR      WATKO
F174 AF                 WATK1:  XOR     A
F175 77                         LD      (HL),A
F176 C9                         RET
F177                    ;
F177                    ; WAIT TIMER
F177                    ;
F177 010600             WATIM:  LD      BC,0006H
F17A 0B                 WATIO:  DEC     BC
F17B 78                         LD      A,B
F17C B1                         OR      C
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 29              84.06.26

F17D C8                      RET     Z
F17E                 ;
F17E                 ; KEY SEARCH
F17E                 ;
F17E 3E01                    LD      A,01H
F180 CDA1F1                  CALL    SRCKO
F183 FE7F                    CP      7FH               ;'Y' ?
F185 37                      SCF
F186 C8                      RET     Z
F187 3E03                    LD      A,03H
F189 CDA1F1                  CALL    SRCKO
F18C FEFB                    CP      FBH               ;'N' ?
F18E 2809                    JR      Z,WATI1
F190 3E08                    LD      A,08H
F192 CDA1F1                  CALL    SRCKO
F195 FE7E                    CP      7EH               ;'BREAK+SHIFT' ?
F197 20E1                    JR      NZ,WATIO
F199                 ;
F199                 ; JUMP ST1X
F199                 ;
F199 AF              WATI1:  XOR     A
F19A 77                      LD      (HL),A
F19B 31F010                  LD      SP,10F0H
F19E C3A7EA                  JP      ST1X
F1A1                 ;
F1A1                 ; KEY SCAN
F1A1                 ;
F1A1                 SRCKO:  ENT
F1A1 E5                      PUSH    HL
F1A2 2100E0                  LD      HL,E000H
F1A5 77                      LD      (HL),A
F1A6 23                      INC     HL
F1A7 7E              SRCK2:  LD      A,(HL)
F1A8 F5                      PUSH    AF
F1A9 C5                      PUSH    BC
F1AA 0614                    LD      B,14H             ;1ms*20=20ms
F1AC CD1BF7          SRCK1:  CALL    DLY1M             ;1ms DELAY
F1AF 10FB                    DJNZ    SRCK1
F1B1 C1                      POP     BC
F1B2 F1                      POP     AF
F1B3 BE                      CP      (HL)
F1B4 20F1                    JR      NZ,SRCK2
F1B6 E1                      POP     HL
F1B7 C9                      RET
F1B8                 ;
F1B8                 ;==============================
F1B8                 ;
F1B8                 ;    QUICK DISK COPY COMMAND
F1B8                 ;
F1B8                 ;==============================
F1B8                 ;
F1B8                 QC:     ENT
F1B8 CD7AEF                  CALL    IOFRS             ;IOCS FLAG RESET
F1BB CDB5EF                  CALL    QDRCK             ;READY CHECK
F1BE 382F                    JR      C,QER06
F1C0 CD16ED                  CALL    FNINP             ;INPUT FILE NAME
F1C3 3AA311                  LD      A,(BUFF)
F1C6 FE0D                    CP      0DH
F1C8 28EE                    JR      Z,QC
F1CA CD18F3                  CALL    HDPCL             ;HD POINT CLEAR
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 30            84.06.26

F1CD 11FBEE                LD     DE,MSGLD         ;'Loading'
F1D0 DF                    RST    3
F1D1                ;
F1D1                ; FILE SEARCH
F1D1                ;
F1D1 CD85EF                CALL   FILSCH
F1D4 3819                  JR     C,QER06
F1D6                ;
F1D6                ; DISP 'LOADING.....'
F1D6                ;
F1D6 11F110                LD     DE,NAME
F1D9 DF                    RST    3
F1DA                ;
F1DA                ; IOCS PARAMETER SET
F1DA                ;
F1DA 210012                LD     HL,1200H
F1DD 223211                LD     (QDPC),HL
F1E0 2A0411                LD     HL,(QSIZE)
F1E3 223411                LD     (QDPE),HL
F1E6 210301                LD     HL,0103H         ;READ DATA BLOCK CMD.
F1E9 223011                LD     (QDPA),HL
F1EC                ;
F1EC                ; READ DATA BLOCK
F1EC                ;
F1EC CD0DE8                CALL   QDIOS
F1EF 3821           QER06: JR     C,QER07
F1F1                ;
F1F1                ; BELL
F1F1                ;
F1F1 CD3E00                CALL   BELL
F1F4                ;
F1F4                ; DISP 'SET DISK.....'
F1F4                ;
F1F4 CD0900                CALL   NL
F1F7 114CF4                LD     DE,QDQCM         ;'SET DISTINATION DISK'
F1FA DF                    RST    3
F1FB                ;
F1FB                ; ERROR RETURN SET
F1FB                ;
F1FB 3E02           QCSPST: LD    A,02H
F1FD 323A11                LD     (QDCPA),A
F200 21FBF1                LD     HL,QCSPST
F203 31EE10                LD     SP,10EEH
F206 E3                    EX     (SP),HL
F207                ;
F207                ; WAIT 'Y'
F207                ;
F207 CD5AF1                CALL   WAITY            ;'OK?(Y/N)'
F20A                ;
F20A                ; READY & WPRT CHECK
F20A                ;
F20A CD79F0                CALL   QDRWCK
F20D 3803                  JR     C,QER07
F20F                ;
F20F                ; FILE END SEARCH
F20F                ;
F20F CD37F0                CALL   FEDSCH           ;FILE END SEARCH
F212 DABBF2         QER07: JP     C,QERTRT         ;ALREADY EXIST ERR
F215 FE28                  CP     40               ;NOT FOUND ?
F217 C2BBF2                JP     NZ,QERTRT
```

```
** Z80 ASSEMBLER ZZ-004C  <9Z-502M>  PAGE 31              84.06.26

F21A                  ;
F21A                  ; SAVE FILE START
F21A                  ;
F21A 210012                     LD      HL,1200H          ;DATA ADDR
F21D C30CF0                     JP      QSVIOS            ;JUMP SAVE ROUTIN
F220                  ;
F220                  ; NON KEY CHECK
F220                  ;
F220                  NKYWT:    ENT
F220 060A                       LD      B,0AH
F222 2100E0           NKYWTO:   LD      HL,E000H
F225 05                         DEC     B
F226 70                         LD      (HL),B
F227 04                         INC     B
F228 23                         INC     HL
F229 7E                         LD      A,(HL)
F22A FEFF                       CP      FFH
F22C 20F2                       JR      NZ,NKYWT
F22E 10F2                       DJNZ    NKYWTO
F230 C9                         RET
F231                  ;
F231                  ;=============================
F231                  ;
F231                  ;    QUICK DISK XFER COMMAND
F231                  ;
F231                  ;=============================
F231                  ;
F231                  QX:       ENT
F231                  ;
F231                  ; LOADING FROM CMT
F231                  ;
F231 CDD804                     CALL    ?RDI
F234 3818                       JR      C,QXST1
F236 2A0411                     LD      HL,(DTADR)        ;DATA ADRS
F239                  ;
F239                  ; DISP 'Loading...'
F239                  ;
F239 CD0900                     CALL    NL
F23C 11FBEE                     LD      DE,MSGLD          ;'Loading'
F23F DF                         RST     3
F240 11F110                     LD      DE,NAME
F243 DF                         RST     3
F244                  ;
F244                  ; READ DATA  (1200H~)
F244                  ;
F244 010012                     LD      BC,1200H
F247 ED430411                   LD      (DTADR),BC
F24B CDF804                     CALL    ?RDD
F24E DA4CEB           QXST1:    JP      C,?ERX
F251 220411                     LD      (DTADR),HL
F254                  ;
F254                  ; BELL
F254                  ;
F254 CD3E00                     CALL    BELL
F257                  ;
F257                  ; INPUT FILE NAME
F257                  ;
F257 CD16ED           QXFNO:    CALL    FNINP             ;INPUT FILENAME
F25A 3AA311                     LD      A,(BUFF)
F25D FE0D                       CP      0DH
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 32               84.06.26

F25F 28F6                    JR     Z,QXFNO
F261 21A311                  LD     HL,BUFF
F264 11F110                  LD     DE,NAME
F267 011100                  LD     BC,0011H
F26A EDB0                    LDIR
F26C                ;
F26C                ; DISP 'SET DISK.....'
F26C                ;
F26C CD0900                  CALL   NL
F26F 114CF4                  LD     DE,QDQCM         ;'Set distination disk'
F272 DF                      RST    3
F273                ;
F273                ; ATRB CHANGE (700→1500)
F273                ;
F273 3AF010                  LD     A,(ATRB)
F276 FE04                    CP     04H              ;700 BSD FILE ?
F278 2806                    JR     Z,ATRCO
F27A FE05                    CP     05H              ;700 BTX FILE ?
F27C 2006                    JR     NZ,QXSPST
F27E 3D                      DEC    A
F27F 3D                      DEC    A
F280 3D             ATRCO:   DEC    A
F281 32F010                  LD     (ATRB),A         ;ATRB SET
F284                ;
F284                ; ERROR RETURN SET
F284                ;
F284 3E02           QXSPST:  LD     A,02H
F286 323A11                  LD     (QDCPA),A
F289 2184F2                  LD     HL,QXSPST
F28C 31EE10                  LD     SP,10EEH
F28F E3                      EX     (SP),HL
F290                ;
F290                ; WAIT 'Y'
F290                ;
F290 CD5AF1                  CALL   WAITY            ;'OK?(Y/N)'
F293                ;
F293                ; PARAMETER TRANS
F293                ;
F293 212E11                  LD     HL,112EH
F296 113011                  LD     DE,1130H
F299 012D00                  LD     BC,002DH
F29C EDB8                    LDDR
F29E 210000                  LD     HL,0000H
F2A1 220211                  LD     (1102H),HL
F2A4                ;
F2A4                ; IOCS FLSG RESET
F2A4                ;
F2A4 CD7AEF                  CALL   IOFRS
F2A7                ;
F2A7                ; READY & WPRT CHECK
F2A7                ;
F2A7 CD79F0                  CALL   QDRWCK
F2AA 3803                    JR     C,QERO8
F2AC                ;
F2AC                ; FILE END SEARCH
F2AC                ;
F2AC CD37F0                  CALL   FEDSCH
F2AF 380A           QERO8:   JR     C,QERTRT
F2B1                ;
F2B1                ; ERROR CHECK
```

```
**  Z80 ASSEMBLER ZZ-004C  <9Z-502M>  PAGE 33               84.06.26

F2B1                      ;
F2B1 FE28                    CP      NTFECD
F2B3 2006                    JR      NZ,QERTRT
F2B5                      ;
F2B5                      ; SAVE ROUTIN JUMP
F2B5                      ;
F2B5 210012                  LD      HL,1200H          ;DATA ADDR
F2B8 C30CF0                  JP      QSVIOS
F2BB                      ;
F2BB                      ;====================
F2BB                      ;
F2BB                      ;    ERROR TREATMENT
F2BB                      ;
F2BB                      ;====================
F2BB                      ;
F2BB                      QERTRT: ENT
F2BB 1188F3                  LD      DE,MGNFE          ;'NOT FOUND ERR'
F2BE FE28                    CP      NTFECD            ;NOT FOUND ERR
F2C0 283B                    JR      Z,QERMF
F2C2 1121F4               QERTO:  LD      DE,MGBDE     ;'BAD DISK ERROR'
F2C5 FE39                    CP      BDSKCD            ;BAD DISK ERR
F2C7 2834                    JR      Z,QERMF
F2C9 11D6F3               QER1:   LD      DE,MGWPT     ;'WRITE PROTECT'
F2CC FE2E                    CP      WPRTCD            ;WRITE PROTECT
F2CE 282D                    JR      Z,QERMF
F2D0 11E9F3               QER2:   LD      DE,MGNRE     ;'NOT READY'
F2D3 FE32                    CP      QNTRCD            ;NOT READY
F2D5 2826                    JR      Z,QERMF
F2D7 11F8F3               QER3:   LD      DE,MGNSE     ;'NO FILE SPACE ERROR'
F2DA FE35                    CP      NFSECD            ;NO FILE SPACE ERR
F2DC 281F                    JR      Z,QERMF
F2DE 110FF4               QER4:   LD      DE,MGUFE     ;'UNFORMAT'
F2E1 FE36                    CP      UNFMCD            ;UNFORMAT ERR
F2E3 2818                    JR      Z,QERMF
F2E5 11C2F3               QER5:   LD      DE,MGALE     ;'ALREADY EXIST'
F2E8 FE2A                    CP      ALEXCD            ;ALREADY EXIST
F2EA 2811                    JR      Z,QERMF
F2EC 119CF3               QER6:   LD      DE,MGTME     ;'TOO MANY FILES'
F2EF FE33                    CP      TMFECD            ;TOO MANY FILES ERR
F2F1 280A                    JR      Z,QERMF
F2F3 1133F4               QER7:   LD      DE,MGBRK     ;'BREAK!'
F2F6 FE00                    CP      BRKCD             ;BREAK?
F2F8 2803                    JR      Z,QERMF
F2FA 11B4F3               QER8:   LD      DE,MGHDE     ;'HARD ERROR'
F2FD                      ;
F2FD                      ; MOTOR OFF
F2FD                      ;
F2FD 3E06                 QERMF:  LD      A,06H        ;MOTOR OFF CMD.
F2FF 323011                  LD      (QDPA),A
F302 CD0DEB                  CALL    QDIOS
F305 CD18F3                  CALL    HDPCL
F308                      ;
F308 3A3A11                  LD      A,(QDCPA)
F30B 1F                      RRA
F30C D8                      RET     C                 ;BOOT ERR
F30D F5                      PUSH    AF
F30E CD0900                  CALL    NL
F311 DF                      RST     3
F312 F1                      POP     AF
F313 1F                      RRA
```

```
** Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 34                        84.06.26

F314 D8                      RET    C                   ;CMD QC,QX ERR
F315 C3A7EA                  JP     ST1X
F318                ;
F318                ; HEADER POINT CLEAR
F318                ;
F318                HDPCL:   ENT
F318 3E05                    LD     A,05H               ;HD POINT CLEAR CMD.
F31A 323011                  LD     (QDPA),A
F31D CD0DE8                  CALL   QDIOS
F320 C9                      RET
F321                ;
F321                ;===================
F321                ;
F321                ;   MESSAGE TABLE
F321                ;
F321                ;===================
F321                ;
F321 4F4B21         MSGQOK:  DEFM   'OK!'
F324 0D                      DEFB   0DH
F325 20202020       MSGQ01:  DEFM   '    OBJ   '
F329 4F424A20
F32D 2020
F32F 0D                      DEFB   0DH
F330 20202020       MSGQ02:  DEFM   '    BTX   '
F334 42545820
F338 2020
F33A 0D                      DEFB   0DH
F33B 20202020       MSGQ03:  DEFM   '    BSD   '
F33F 42534420
F343 2020
F345 0D                      DEFB   0DH
F346 20202020       MSGQ04:  DEFM   '    BRD   '
F34A 42524420
F34E 2020
F350 0D                      DEFB   0DH
F351 20202020       MSGQ05:  DEFM   '    RB    '
F355 52422020
F359 2020
F35B 0D                      DEFB   0DH
F35C 20202020       MSGQ07:  DEFM   '    LIB   '
F360 4C494220
F364 2020
F366 0D                      DEFB   0DH
F367 20202020       MSGQ10:  DEFM   '    SYS   '
F36B 53595320
F36F 2020
F371 0D                      DEFB   0DH
F372 20202020       MSGQ11:  DEFM   '    GR    '
F376 47522020
F37A 2020
F37C 0D                      DEFB   0DH
F37D 20202020       MSGQ??:  DEFM   '    ???   '
F381 3F3F3F20
F385 2020
F387 0D                      DEFB   0DH
F388 51443A46       MGNFE:   DEFM   'QD:F■ILE NOT FOUND■'
F38C 05494C45
F390 204E4F54
F394 20464F55
F398 4E4405
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 35           84.06.26

F39B OD                         DEFB    ODH
F39C 51443A54          MGTME:   DEFM    'QD:T□OD MANY FILES ERR□'
F3A0 054F4F20
F3A4 4D414E59
F3A8 2046494C
F3AC 45532045
F3B0 525205
F3B3 OD                         DEFB    ODH
F3B4 51443A48          MGHDE:   DEFM    'QD:H□ARD ERR□'
F3B8 05415244
F3BC 20455252
F3C0 05
F3C1 OD                         DEFB    ODH
F3C2 41054C52          MGALE:   DEFM    'A□LREADY EXIST ERR□'
F3C6 45414459
F3CA 20455849
F3CE 53542045
F3D2 525205
F3D5 OD                         DEFB    ODH
F3D6 51443A57          MGWPT:   DEFM    'QD:W□RITE PROTECT□'
F3DA 05524954
F3DE 45205052
F3E2 4F544543
F3E6 5405
F3E8 OD                         DEFB    ODH
F3E9 51443A4E          MGNRE:   DEFM    'QD:N□OT READY□'
F3ED 054F5420
F3F1 52454144
F3F5 5905
F3F7 OD                         DEFB    ODH
F3F8 51443A4E          MGNSE:   DEFM    'QD:N□O FILE SPACE ERR□'
F3FC 054F2046
F400 494C4520
F404 53504143
F408 45204552
F40C 5205
F40E OD                         DEFB    ODH
F40F 51443A55          MGUFE:   DEFM    'QD:U□NFORMAT ERR□'
F413 054E464F
F417 524D4154
F41B 20455252
F41F 05
F420 OD                         DEFB    ODH
F421 51443A42          MGBDE:   DEFM    'QD:B□AD DISK ERR□'
F425 05414420
F429 4449534B
F42D 20455252
F431 05
F432 OD                         DEFB    ODH
F433                   MGBRK:   ENT
F433 42055245                   DEFM    'B□REAK□!'
F437 414B0521
F43B OD                         DEFB    ODH
F43C 51443A46          QDFMG:   DEFM    'QD:F□ORMATTING□'
F440 054F524D
F444 41545449
F448 4E4705
F44B OD                         DEFB    ODH
F44C 53054554          QDQCM:   DEFM    'S□ET DESTINATION DISK□'
F450 20444553
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 36              84.06.26

F454 54494E41
F458 54494F4E
F45C 20444953
F460 4B05
F462 0D                       DEFB   0DH
F463 4F4B3F28        QDQCMO:  DEFM   ´OK?(Y/N)´
F467 592F4E29
F46B 0D                       DEFB   0DH
F46C 44054952        DIRMSG:  DEFM   ´D□IRECTORY OF□ QD:´
F470 4543544F
F474 5259204F
F478 46052051
F47C 443A
F47E 0D                       DEFB   0DH
F47F                          SKP    H
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 37              84.06.26

F47F                      ;
F47F                      ;----< MFM MINIFLOPPY CONTROL >----
F47F                      ;         SUBROUTINE  (MB8876)
F47F                      ;
F47F                      ;
F47F                      ;
F47F                      ;    CALL CONDITION
F47F                      ;
F47F                      ;    CASE OF DISK INITIALIZE
F47F                      ;       DRIVE No=IX+0   (0~3)
F47F                      ;
F47F                      ;    CASE OF SEQUENTIALIZE READ & WRITE
F47F                      ;       DRIVE No=IX+0   (0~3)
F47F                      ;
F47F                      ;       SECTOR ADDRESS=IX+1,2 (0-045FH)
F47F                      ;                              (0-1119 )
F47F                      ;       BYTE SIZE    =IX+3,4
F47F                      ;       ADDRESS      =IX+5,6
F47F                      ;       NEXT TRACK   =IX+7
F47F                      ;       NEXT SECTOR  =IX+8
F47F                      ;       START TRACK  =IX+9
F47F                      ;       START SECTOR =IX+10
F47F                      ;
F47F                      ;
F47F                      ; I/O PORT ADDRESS
F47F                      ;
00D8  P                   CR:      EQU    D8H
00D9  P                   TR:      EQU    D9H
00DA  P                   SCR:     EQU    DAH
00DB  P                   DR:      EQU    DBH
00DC  P                   DM:      EQU    DCH
00DD  P                   HS:      EQU    DDH
F47F                      ;
F47F                      ;   SUBROUTINE WORK
F47F                      ;
CF00  P                   BPRO:    EQU    CF00H            ;IFM LOADING ADDR.
11A3  P                   BUF:     EQU    11A3H
CEE9  P                   BPARA:   EQU    BPRO-23          ;FD WORK
F47F                      ;
F47F                      ;
F47F                      ;--------< ERCODE MAP >--------
F47F                      ;
F47F                      ; 50 :NOT READY
F47F                      ; 41 :DATA ERROR
F47F                      ;                  TRACK 00 ERR
F47F                      ;                  WRITE PROTECT ERR
F47F                      ;                  SEEK ERR
F47F                      ;                  CRC ERR
F47F                      ;                  LOST DATA
F47F                      ; 54 :UNFORMAT
F47F                      ;        RECORD NOT FOUND
F47F                      ; 56 :INVALID DATA
F47F                      ;
F47F                      ;
F47F                      ;
F47F                      ;  MAIN ROUTINE
F47F                      FDX:     ENT
F47F E3                            EX     (SP),HL
F480 22FECE                        LD     (BPARA+21),HL
F483 CD18E9                        CALL   FDCC&            ;FD I/O CHECK
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 38          84.06.26

F486 C21BF5              JP      NZ,NOTIO
F489 11E9CE              LD      DE,BPARA
F48C 2132F5              LD      HL,BOOT
F48F 010B00              LD      BC,11
F492 EDB0                LDIR
F494 CD91F5              CALL    MOFF
F497 DD21E9CE   SJP:     LD      IX,BPARA
F49B CD08F6              CALL    BREAD
F49E 2100CF              LD      HL,BPRO
F4A1 112BF5              LD      DE,IPLMC
F4A4 0607                LD      B,7
F4A6 4E         MCHECK:  LD      C,(HL)
F4A7 1A                  LD      A,(DE)
F4A8 B9                  CP      C
F4A9 C20FF5              JP      NZ,MASTE
F4AC 23                  INC     HL
F4AD 13                  INC     DE
F4AE 10F6                DJNZ    MCHECK
F4B0 11D4ED              LD      DE,IPLMO           ;'IPL IS LOADING'
F4B3 DF                  RST     3
F4B4 1107CF              LD      DE,BPRO+7          ;NAME
F4B7 DF                  RST     3
F4B8 2A16CF              LD      HL,(BPRO+16H)      ;DATA ADDR
F4BB 7C                  LD      A,H
F4BC B5                  OR      L
F4BD 2007                JR      NZ,FDX1
F4BF 2A18CF              LD      HL,(BPRO+18H)      ;EXEC ADDR
F4C2 7C                  LD      A,H
F4C3 B5                  OR      L
F4C4 2806                JR      Z,FDX2
F4C6 AF         FDX1:    XOR     A
F4C7 2A16CF              LD      HL,(BPRO+16H)      ;DATA ADDR
F4CA 1805                JR      FDX3
F4CC 3EFF       FDX2:    LD      A,FFH
F4CE 210012              LD      HL,1200H
F4D1 32FDCE     FDX3:    LD      (BPARA+20),A
F4D4 DD7505              LD      (IX+5),L
F4D7 DD7406              LD      (IX+6),H
F4DA 2A14CF              LD      HL,(BPRO+14H)      ;BYTE SIZE
F4DD DD7503              LD      (IX+3),L
F4E0 DD7404              LD      (IX+4),H
F4E3 2A1ECF              LD      HL,(BPRO+1EH)      ;START SECTOR
F4E6 DD7501              LD      (IX+1),L
F4E9 DD7402              LD      (IX+2),H
F4EC            ;
F4EC CD08F6              CALL    BREAD
F4EF CD91F5              CALL    MOFF
F4F2 3AFDCE              LD      A,(BPARA+20)
F4F5 FEFF                CP      FFH
F4F7 200F                JR      NZ,FDEXC
F4F9 D3E0                OUT     (E0H),A
F4FB 210012              LD      HL,1200H
F4FE ED5B16CF            LD      DE,(BPRO+16H)      ;DATA ADDR
F502 ED4B14CF            LD      BC,(BPRO+14H)      ;BYTE SIZE
F506 EDB0                LDIR
F508 2A18CF     FDEXC:   LD      HL,(BPRO+18H)      ;EXEC ADDR
F50B 010002              LD      BC,0200H           ;FLOPPY DISK CORD
F50E E9                  JP      (HL)
F50F CD91F5     MASTE:   CALL    MOFF
F512 114EEE              LD      DE,ERRM1           ;'NOT MASTER'
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 39              84.06.26

F515 180C                  JR      ERRTR1
F517                   ;
F517 FE32          ERRTRT: CP      50
F519 2005                  JR      NZ,ERRTRO
F51B 1107EE        NOTIO:  LD      DE,IPLM3             ;'MAKE READY FD'
F51E 1803                  JR      ERRTR1
F520 113BEE        ERRTRO: LD      DE,ERRMO             ;'FD:L■OADING ERROR■'
F523 31EE10        ERRTR1: LD      SP,10EEH
F526 2AFECE                LD      HL,(BPARA+21)
F529 E3                    EX      (SP),HL
F52A C9                    RET
F52B                   ;
F52B                   ;    PARAMETER SETTING
F52B                   ;
F52B 03            IPLMC:  DEFB    03H                  ;IPL MASTER FLAG
F52C 49504C50              DEFM    'IPLPRO'
F530 524F
F532 00            BOOT:   DEFB    0                    ;DRIVE NO.
F533 0000                  DEFW    0000H                ;SECTOR ADDR.
F535 0001                  DEFW    0100H                ;IFM BYTE SIZE
F537 00CF                  DEFW    BPRO                 ;IFM LOADING ADDR.
F539 0000                  DEFW    0                    ;IX+7,8
F53B 0000                  DEFW    0                    ;IX+9,10
F53D                   ;
F53D                   ; READY CHECK
F53D                   ;
F53D               READY:  ENT
F53D 3AF5CE                LD      A,(MTFG)
F540 0F                    RRCA
F541 D478F5                CALL    NC,FDMTON
F544 DD7E00                LD      A,(IX+0)             ;DRIVE NO GET
F547 F684                  OR      84H
F549 D3DC                  OUT     (DM),A               ;DRIVE SELECT MOTON
F54B AF                    XOR     A
F54C 32F4CE                LD      (CMD),A
F54F CD22F7                CALL    DLY60M
F552 210000                LD      HL,0
F555 2B            REDYO:  DEC     HL
F556 7C                    LD      A,H
F557 B5                    OR      L
F558 2819                  JR      Z,REDY1
F55A DBD8                  IN      A,(CR)               ;STATUS GET
F55C 2F                    CPL
F55D 07                    RLCA
F55E 38F5                  JR      C,REDYO
F560 DD4E00                LD      C,(IX+0)
F563 21F6CE                LD      HL,CLBFO
F566 0600                  LD      B,0
F568 09                    ADD     HL,BC
F569 CB46                  BIT     0,(HL)
F56B 2005                  JR      NZ,REDY2
F56D CDA9F5                CALL    RCLB
F570 CBC6                  SET     0,(HL)
F572 C9            REDY2:  RET
F573 3E32          REDY1:  LD      A,50
F575 C3F1F6                JP      ERJMP
F578                   ;
F578                   ; MOTOR ON
F578                   ;
F578               FDMTON: ENT
```

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 40              84.06.26

F578 3E80                      LD      A,80H
F57A D3DC                      OUT     (DM),A
F57C 0610                      LD      B,16
F57E CD22F7           MTD1:    CALL    DLY60M
F581 10FB                      DJNZ    MTD1
F583 3E01                      LD      A,1
F585 32F5CE                    LD      (MTFG),A
F588 C9                        RET
F589                  ;
F589                  ; SEEK TREATMENT
F589                  ;
F589                  SEEK:    ENT
F589 3E1B                      LD      A,1BH
F58B CDB6F5                    CALL    CMDOT1
F58E E699                      AND     99H
F590 C9                        RET
F591                  ;
F591                  ;  MOTER OFF
F591                  ;
F591                  MOFF:    ENT
F591 F5                        PUSH    AF
F592 CD1BF7                    CALL    DLY1M               ;1000 US DELAY
F595 AF                        XOR     A
F596 D3DC                      OUT     (DM),A
F598 32F6CE                    LD      (CLBF0),A
F59B 32F7CE                    LD      (CLBF1),A
F59E 32F8CE                    LD      (CLBF2),A
F5A1 32F9CE                    LD      (CLBF3),A
F5A4 32F5CE                    LD      (MTFG),A
F5A7 F1                        POP     AF
F5A8 C9                        RET
F5A9                  ;
F5A9                  ; RECALIBLATION
F5A9                  ;
F5A9                  RCLB:    ENT
F5A9 3E0B                      LD      A,0BH
F5AB CDB6F5                    CALL    CMDOT1
F5AE E685                      AND     85H
F5B0 EE04                      XOR     04
F5B2 C8                        RET     Z
F5B3 C3C2F6                    JP      ERROR               ;
F5B6                  ;
F5B6                  ;     COMAND OUT ROUTINE
F5B6                  ;
F5B6 32F4CE           CMDOT1:  LD      (CMD),A
F5B9 2F                        CPL
F5BA D3D8                      OUT     (CR),A
F5BC CDC9F5                    CALL    BSYON
F5BF CD22F7                    CALL    DLY60M
F5C2 DBD8                      IN      A,(CR)
F5C4 2F                        CPL
F5C5 32FBCE                    LD      (STAFG),A
F5C8 C9                        RET
F5C9                  ;
F5C9                  ; BUSY AND WAIT
F5C9                  ;
F5C9 D5               BSYON:   PUSH    DE
F5CA E5                        PUSH    HL
F5CB CD02F6                    CALL    BSY0
F5CE 210000           BSYON2:  LD      HL,0000H
```

IPLのリスト

```
**   Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 41              84.06.26

F5D1 2B              BSYONO: DEC     HL
F5D2 7C                      LD      A,H
F5D3 B5                      OR      L
F5D4 2808                    JR      Z,BSYON1
F5D6 DBD8                    IN      A,(CR)
F5D8 0F                      RRCA
F5D9 30F6            BSYON3: JR      NC,BSYONO
F5DB E1                      POP     HL
F5DC D1                      POP     DE
F5DD C9                      RET
F5DE                 ;
F5DE 1D              BSYON1: DEC     E
F5DF 20ED                    JR      NZ,BSYON2
F5E1 3E29            BSYONE: LD      A,41
F5E3 E1                      POP     HL
F5E4 D1                      POP     DE
F5E5 C3F1F6                  JP      ERJMP
F5E8                 ;
F5E8 D5              BSYOFF: PUSH    DE
F5E9 E5                      PUSH    HL
F5EA CD02F6                  CALL    BSYO
F5ED 210000          BSYOF2: LD      HL,0000H
F5F0 2B              BSYOFO: DEC     HL
F5F1 7C                      LD      A,H
F5F2 B5                      OR      L
F5F3 2808                    JR      Z,BSYOF1
F5F5 DBD8                    IN      A,(CR)
F5F7 0F                      RRCA
F5F8 38F6            BSYOF3: JR      C,BSYOFO
F5FA E1                      POP     HL
F5FB D1                      POP     DE
F5FC C9                      RET
F5FD                 ;
F5FD 1D              BSYOF1: DEC     E
F5FE 20ED                    JR      NZ,BSYOF2
F600 18DF            BSYOFE: JR      BSYONE
F602                 ;
F602 CD14F7          BSYO:   CALL    DLY80U
F605 1E07                    LD      E,07H
F607 C9                      RET
F608                 ;
F608                 ;
F608                 ; SEQENTIAL READ
F608                 ;
F608                 BREAD:  ENT
F608 CDF7F6                  CALL    CNVRT
F60B CD5CF6          READ1:  CALL    PARST1
F60E CD7CF6          REB:    CALL    SIDST
F611 CD89F5                  CALL    SEEK
F614 C2F1F6                  JP      NZ,ERJMP
F617 CD8CF6                  CALL    PARST2
F61A F3                      DI
F61B 3E94                    LD      A,94H          ;READ & CMD
F61D CDAFF6                  CALL    CMDOT2
F620 0600            RE6:    LD      B,00
F622 DBD8            RE4:    IN      A,(CR)
F624 0F                      RRCA
F625 381A                    JR      C,RE3
F627 0F                      RRCA
F628 38F8                    JR      C,-6
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 42              84.06.26

F62A EDA2                    INI
F62C 20F4                    JR      NZ,RE4
F62E DD3408                  INC     (IX+8)
F631 DD7E08                  LD      A,(IX+8)
F634 FE11                    CP      17
F636 2805                    JR      Z,+7
F638 15                      DEC     D
F639 20E5                    JR      NZ,RE6
F63B 1801                    JR      +3
F63D 15                      DEC     D
F63E CDB9F6                  CALL    INTER
F641 FB             RE3:     EI
F642 DBD8                    IN      A,(CR)
F644 2F                      CPL
F645 32FBCE                  LD      (STAFG),A
F648 E6FF                    AND     FFH
F64A 2076                    JR      NZ,ERROR
F64C CD9DF6                  CALL    ADJ
F64F CA57F6                  JP      Z,REND
F652 DD7E07                  LD      A,(IX+7)
F655 18B7                    JR      RE8
F657 3E80           REND:    LD      A,80H
F659 D3DC                    OUT     (DM),A
F65B C9                      RET
F65C                ;
F65C                ;
F65C                ;
F65C                ;  PARAMETER SET
F65C                ;
F65C                ;
F65C CD3DF5         PARST1:  CALL    READY
F65F DD5604                  LD      D,(IX+4)
F662 DD7E03                  LD      A,(IX+3)
F665 B7                      OR      A
F666 2801                    JR      Z,+3
F668 14                      INC     D
F669 DD7E0A                  LD      A,(IX+10)
F66C DD7708                  LD      (IX+8),A
F66F DD7E09                  LD      A,(IX+9)
F672 DD7707                  LD      (IX+7),A
F675 DD6E05                  LD      L,(IX+5)
F678 DD6606                  LD      H,(IX+6)
F67B C9                      RET
F67C                ;
F67C                ;  SIZE SEEK SET
F67C                ;
F67C CB3F           SIDST:   SRL     A
F67E 2F                      CPL
F67F D3DB                    OUT     (DR),A
F681 3004                    JR      NC,+6
F683 3E01                    LD      A,1
F685 1801                    JR      +3
F687 AF                      XOR     A
F688 2F                      CPL
F689 D3DD                    OUT     (HS),A
F68B C9                      RET
F68C                ;
F68C                ;
F68C                ;  TRACK & SECTOR SET
F68C                ;
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 43                 84.06.26

F68C 0EDB              PARST2: LD      C,DBH
F68E DD7E07                    LD      A,(IX+7)
F691 CB3F                      SRL     A
F693 2F                        CPL
F694 D3D9                      OUT     (TR),A
F696 DD7E08                    LD      A,(IX+8)
F699 2F                        CPL
F69A D3DA                      OUT     (SCR),A
F69C C9                        RET
F69D                   ;
F69D                   ;
F69D                   ; ADJUST SECT & TRACK
F69D                   ;
F69D DD7E08            ADJ:    LD      A,(IX+8)
F6A0 FE11                      CP      17
F6A2 2008                      JR      NZ,+10
F6A4 3E01                      LD      A,1
F6A6 DD7708                    LD      (IX+8),A
F6A9 DD3407                    INC     (IX+7)
F6AC 7A                        LD      A,D
F6AD B7                        OR      A
F6AE C9                        RET
F6AF                   ;
F6AF                   ;  COMAMD OUT & WAIT
F6AF                   ;
F6AF 32F4CE            CMDOT2: LD      (CMD),A
F6B2 2F                        CPL
F6B3 D3D8                      OUT     (CR),A
F6B5 CDE8F5                    CALL    BSYOFF
F6B8 C9                        RET
F6B9                   ;
F6B9                   ;
F6B9                   ;  FORCE INTERRUPT
F6B9                   ;
F6B9 3ED8              INTER:  LD      A,D8H
F6BB 2F                        CPL
F6BC D3D8                      OUT     (CR),A
F6BE CDC9F5                    CALL    BSYON
F6C1 C9                        RET
F6C2                   ;
F6C2                   ;
F6C2                   ; STATUS CHECK
F6C2                   ;
F6C2 3AF4CE            ERROR:  LD      A,(CMD)
F6C5 FE0B                      CP      0BH
F6C7 281B                      JR      Z,ERCK1
F6C9 FE1B                      CP      1BH
F6CB 2817                      JR      Z,ERCK1
F6CD FEF4                      CP      F4H
F6CF 2813                      JR      Z,ERCK1
F6D1 3AFBCE                    LD      A,(STAFG)
F6D4 CB7F                      BIT     7,A
F6D6 2017                      JR      NZ,ERRET
F6D8 CB77                      BIT     6,A
F6DA 200F                      JR      NZ,ERRET1
F6DC CB67                      BIT     4,A
F6DE 3E36                      LD      A,54
F6E0 200F                      JR      NZ.ERJMP
F6E2 1807                      JR      ERRET1
F6E4 3AFBCE            ERCK1:  LD      A,(STAFG)
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 44            84.06.26

F6E7 CB7F                   BIT     7,A
F6E9 2004                   JR      NZ,ERRET
F6EB 3E29          ERRET1:  LD      A,41
F6ED 1802                   JR      ERJMP
F6EF 3E32          ERRET:   LD      A,50
F6F1 CD91F5        ERJMP:   CALL    MOFF
F6F4 C317F5                 JP      ERRTRT
F6F7               ;
F6F7               ;
F6F7               ; SECTOR TO TRACK & SECTOR CONVERT
F6F7               ;
F6F7 0600          CNVRT:   LD      B,0
F6F9 111000                 LD      DE,16
F6FC DD6E01                 LD      L,(IX+1)
F6FF DD6602                 LD      H,(IX+2)
F702 AF                     XOR     A
F703 ED52          TRANS0:  SBC     HL,DE
F705 3803                   JR      C,TRANS1
F707 04                     INC     B
F708 18F9                   JR      TRANS0
F70A 19            TRANS1:  ADD     HL,DE
F70B 60                     LD      H,B
F70C 2C                     INC     L
F70D DD7409                 LD      (IX+9),H
F710 DD750A                 LD      (IX+10),L
F713 C9                     RET
F714               ;
F714               ;
F714               ;  TIME DELAY ( 1M & 60M & 80U )
F714               ;
F714               DLY80U:  ENT
F714 D5                     PUSH    DE
F715 110F00                 LD      DE,15
F718 C326F7                 JP      DLYT
F71B               DLY1M:   ENT
F71B D5                     PUSH    DE
F71C 11A000                 LD      DE,160
F71F C326F7                 JP      DLYT
F722               DLY60M:  ENT
F722 D5                     PUSH    DE
F723 112620                 LD      DE,8230
F726 1B            DLYT:    DEC     DE
F727 7B                     LD      A,E
F728 B2                     OR      D
F729 20FB                   JR      NZ,DLYT
F72B D1                     POP     DE
F72C C9                     RET
F72D               ;
F72D               ;  FLOPPY WORK AREA
F72D               ;
CEF4  P            CMD:     EQU     BPARA+11
CEF5  P            MTFG:    EQU     CMD+1
CEF6  P            CLBF0:   EQU     MTFG+1
CEF7  P            CLBF1:   EQU     CLBF0+1
CEF8  P            CLBF2:   EQU     CLBF1+1
CEF9  P            CLBF3:   EQU     CLBF2+1
CEFA  P            VRFCNT:  EQU     CLBF3+1
CEFB  P            STAFG:   EQU     VRFCNT+1
F72D                        SKP     H
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 45           84.06.26

F72D                    ;
F72D                    ;========================================
F72D                    ;
F72D                    ;     1R12(STATIC RAM BOARD 32Kbyte)
F72D                    ;                     READ/WRITE ROUTINE
F72D                    ;
F72D                    ;========================================
F72D                    ;
1108  P                 COMNT:  EQU     1108H
F72D                    ;
F72D                    ; 1R12 READ SUB.
F72D                    ;
F72D                    1R12RD: ENT
F72D CD34F7                     CALL    1R12RO
F730 C24CEB                     JP      NZ,?ERX
F733 E9                         JP      (HL)
F734                    ;
F734                    1R12RO: ENT
F734                    ;
F734                    ; INFORMATION SUM CHECK
F734                    ;
F734 CDA2F7                     CALL    CHECK
F737 2048                       JR      NZ,12RET
F739                    ;
F739                    ; COUNTER RESET
F739                    ;
F739 ED78                       IN      A,(C)
F73B                    ;
F73B                    ; READ DATA SIZE
F73B                    ;       (COMNT+2)←DATA SIZE
F73B                    ;
F73B 0C                         INC     C               ;C=C+1
F73C ED78                       IN      A,(C)           ;C=C+1
F73E 5F                         LD      E,A
F73F ED78                       IN      A,(C)           ;C=C+1
F741 57                         LD      D,A
F742 ED530A11                   LD      (COMNT+2),DE
F746                    ;
F746                    ; READ DATA ADRS  (HL←DATA ADRS)
F746                    ;
F746 ED78                       IN      A,(C)           ;C=C+1
F748 6F                         LD      L,A
F749 ED78                       IN      A,(C)           ;C=C+1
F74B 67                         LD      H,A
F74C                    ;
F74C                    ; READ EXEC ADRS  (DE←EXEC ADRS)
F74C                    ;
F74C ED78                       IN      A,(C)           ;C=C+1
F74E 5F                         LD      E,A             ; EXEC ADR
F74F ED78                       IN      A,(C)           ;C=C+1
F751 57                         LD      D,A
F752                    ;
F752                    ; READ DATA'S CHECK SUM
F752                    ;       (COMNT+4)←DATA'S CHECK SUM
F752                    ;
F752 D5                         PUSH    DE
F753 ED78                       IN      A,(C)           ;C=C+1
F755 5F                         LD      E,A
F756 ED78                       IN      A,(C)           ;C=C+1
F758 57                         LD      D,A
```

```
** Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 46          84.06.26

F759 ED530C11                LD    (COMNT+4),DE
F75D                  ;
F75D                  ; DE←SIZE
F75D                  ;
F75D ED5B0A11                LD    DE,(COMNT+2)
F761                  ;
F761                  ; READ INFORMATION'S CHECK SUM
F761                  ;       Acc←CHECK SUM
F761                  ;
F761 ED78                    IN    A,(C)            ;C=C+1
F763                  ;
F763                  ; READ DATA BLOCK
F763                  ;
F763 E5                      PUSH  HL
F764 7B                      LD    A,E
F765 B7                      OR    A
F766 2803                    JR    Z,+5
F768 47                      LD    B,A
F769 EDB2                    INIR                   ;C=C+1
F76B 0600                    LD    B,0
F76D 15                      DEC   D
F76E F269F7                  JP    P,-5
F771                  ;
F771                  ; SUM CHECK
F771                  ;
F771 D1                      POP   DE               ;HL←DATA ADRS
F772 ED4B0A11                LD    BC,(COMNT+2)     ;BC←SIZE
F776 CD82F7                  CALL  SUM              ;HL←CHECK SUM
F779 ED5B0C11                LD    DE,(COMNT+4)     ;DE←CHECK SUM DATA
F77D B7                      OR    A
F77E ED52                    SBC   HL,DE            ;OK?
F780 E1                      POP   HL               ;HL←EXEC ADRS
F781 C9               12RET: RET
F782                  ;
F782                  ; CHECK SUM
F782                  ;
F782                  ; IN BC=Size
F782                  ;    DE=Data adr
F782                  ; EXIT HL=Check sum
F782                  ;
F782 C5               SUM:   PUSH  BC
F783 D5                      PUSH  DE
F784 D9                      EXX
F785 210000                  LD    HL,0             ; HL'= CHECK SUM CLR
F788 0E08                    LD    C,8              ; C' = LOOP COUNT
F78A D9                      EXX
F78B 78               SUMCK1: LD   A,B              ; BC = SIZE
F78C B1                      OR    C
F78D 280F                    JR    Z,SUMCK2
F78F 1A                      LD    A,(DE)           ; DE = DATA ADDRESS
F790 D9                      EXX
F791 41                      LD    B,C              ; BC'
F792 07               SUMCK3: RLCA
F793 3001                    JR    NC,+3
F795 23                      INC   HL               ; HL'= CHECK SUM DATA
F796 10FA                    DJNZ  SUMCK3
F798 D9                      EXX
F799 13                      INC   DE               ; DE
F79A 0B                      DEC   BC               ; BC
F79B C38BF7                  JP    SUMCK1
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 47                  84.06.26

F79E D9                 SUMCK2: EXX
F79F D1                         POP     DE
F7A0 C1                         POP     BC
F7A1 C9                         RET
F7A2                    ;
F7A2                    ; INFORMATION'S SUM CHECK
F7A2                    ;
F7A2 ED78               CHECK:  IN      A,(C)           ;COUNTER RESET
F7A4 0608                       LD      B,08H           ;B=Byte Counter
F7A6 1600                       LD      D,00H           ;Sum Counter
F7A8 0C                         INC     C               ;C=C+1
F7A9 ED78               CK1:    IN      A,(C)           ;C=C+1
F7AB C5                         PUSH    BC
F7AC 0608                       LD      B,08H           ;Bit Counter
F7AE 07                 CK2:    RLCA
F7AF 3001                       JR      NC,+3
F7B1 14                         INC     D
F7B2 10FA                       DJNZ    CK2
F7B4 7A                         LD      A,D
F7B5 C1                         POP     BC
F7B6 57                         LD      D,A
F7B7 10F0                       DJNZ    CK1
F7B9 ED78                       IN      A,(C)           ;C=C+1
F7BB 0D                         DEC     C               ;C=C+0
F7BC BA                         CP      D
F7BD C9                         RET
F7BE                    ;
F7BE                    ;==============================
F7BE                    ;
F7BE                    ;       1R12 WRITE SUB.
F7BE                    ;
F7BE                    ;==============================
F7BE                    ;
F7BE                    1R12WR: ENT
F7BE                    ;
F7BE                    ; DISP 'Writing.....'
F7BE                    ;
F7BE 1163F8                     LD      DE,MSGW01       ;'Writing now'
F7C1 CD57F8                     CALL    CRMSGC
F7C4 117AF8                     LD      DE,MSGW02       ;'Master tape set'
F7C7 CD5AF8                     CALL    MSGCR
F7CA                    ;
F7CA                    ; READ INFORMATION BLOCK
F7CA                    ;
F7CA CD2700                     CALL    RDINF
F7CD DA4CEB                     JP      C,?ERX
F7D0                    ;
F7D0                    ; DISP 'Loading.....'
F7D0                    ;
F7D0 11FBEE                     LD      DE,MSGLD        ;'Loading'
F7D3 CD5EF8                     CALL    CRMSG
F7D6 11F110                     LD      DE,NAME
F7D9 DF                         RST     3
F7DA                    ;
F7DA                    ; READ DATA BLOCK
F7DA                    ;
F7DA CD2A00                     CALL    RDDAT
F7DD DA4CEB                     JP      C,?ERX
F7E0                    ;
F7E0                    ; COUNTER RESET
```

```
** Z80 ASSEMBLER 2Z-004C <9Z-502M> PAGE 48                    84.06.26

F7E0                   ;
F7E0 ED78                      IN      A,(C)           ;C=C+0
F7E2                   ;
F7E2                   ; DATA'S SUM CHECK
F7E2                   ;
F7E2 ED430A11                  LD      (COMNT+2),BC    ;C=Port adrs,C=C+0
F7E6 ED5B0411                  LD      DE,(DTADR)
F7EA ED4B0211                  LD      BC,(SIZE)
F7EE D5                        PUSH    DE
F7EF C5                        PUSH    BC
F7F0 CD82F7                    CALL    SUM             ;HL←CHECK SUM
F7F3 220811                    LD      (COMNT),HL
F7F6                   ;
F7F6                   ; WRITE INFORMATION (8Byte)
F7F6                   ;
F7F6 210211                    LD      HL,SIZE
F7F9 ED4B0A11                  LD      BC,(COMNT+2)    ;C=Port adrs,C=C+0
F7FD 0608                      LD      B,08H           ;Byte Counter
F7FF 0C                        INC     C
F800 0C                        INC     C
F801 E5                        PUSH    HL
F802 C5                        PUSH    BC
F803 EDB3                      OTIR                    ;C=C+2
F805 C1                        POP     BC
F806 E1                        POP     HL
F807                   ;
F807                   ; INFORMATION'S SUM CHECK
F807                   ;     Acc←CHECK SUM DATA
F807                   ;
F807 D5                        PUSH    DE              ;DE←SIZE
F808 1600                      LD      D,00H           ;Sum Counter
F80A C5                WCK1:   PUSH    BC
F80B 0608                      LD      B,08H
F80D 7E                        LD      A,(HL)
F80E 07                WCK2:   RLCA
F80F 3001                      JR      NC,WCK3
F811 14                        INC     D
F812 10FA              WCK3:   DJNZ    WCK2
F814 23                        INC     HL
F815 C1                        POP     BC
F816 10F2                      DJNZ    WCK1
F818 7A                        LD      A,D
F819 D1                        POP     DE
F81A ED79                      OUT     (C),A           ;C=C+2
F81C                   ;
F81C                   ; WIRTE DATA BLOCK
F81C                   ;
F81C D1                        POP     DE              ;DE←SIZE
F81D E1                        POP     HL              ;HL←DATA ADRS
F81E 7B                        LD      A,E
F81F B7                        OR      A
F820 2803                      JR      Z,+5
F822 43                        LD      B,E
F823 EDB3                      OTIR                    ;C=C+2
F825 0600                      LD      B,00
F827 15                        DEC     D
F828 F223F8                    JP      P,-5
F82B C3A7EA                    JP      ST1X
F82E                   ;
F82E                   ; 1R12 SET CHECK
```

IPLのリスト

```
F82E                    ;
F82E                    R12CK1: ENT
F82E AF                         XOR    A              ;CARRY FLAG RESET
F82F ED78                       IN     A,(C)          ;COUNTER RESET,C=C+0
F831 0C                         INC    C
F832 ED78                       IN     A,(C)          ;C=C+1
F834 08                         EX     AF,AF'
F835 0D                         DEC    C
F836 ED78                       IN     A,(C)          ;COUNTER RESET
F838 3EA5                       LD     A,A5H
F83A 0C                         INC    C
F83B 0C                         INC    C
F83C ED79                       OUT    (C),A          ;C=C+2
F83E 0D                         DEC    C
F83F 0D                         DEC    C
F840 ED78                       IN     A,(C)          ;COUNTER RESET
F842 0C                         INC    C
F843 ED78                       IN     A,(C)          ;C=C+1
F845 FEA5                       CP     A5H
F847 200B                       JR     NZ,12NSET
F849 0D                         DEC    C
F84A ED78                       IN     A,(C)          ;COUNTER RESET
F84C 08                         EX     AF,AF'
F84D 0C                         INC    C
F84E 0C                         INC    C
F84F ED79                       OUT    (C),A          ;C=C+2
F851 0D                         DEC    C
F852 0D                         DEC    C              ;C=C+0
F853 C9                         RET
F854 AF                 12NSET: XOR    A              ;Z FLAG SET
F855 37                         SCF                   ;NOT SET
F856 C9                         RET
F857                    ;
F857 CD1809             CRMSGC: CALL   0918H
F85A DF                 MSGCR:  RST    3
F85B C31809                     JP     0918H
F85E                    ;
F85E CD1809             CRMSG:  CALL   0918H
F861 DF                         RST    3
F862 C9                         RET
F863                    ;
F863                    ;
F863                    ; MESSAGE TABLE
F863                    ;
F863 5205414D           MSGW01: DEFM   "R□AMBOARD PROGRAMMING□"
F867 424F4152
F86B 44205052
F86F 4F475241
F873 4D4D494E
F877 4705
F879 0D                         DEFB   0DH
F87A 4D054153           MSGW02: DEFM   "M□ASTER TAPE SET□"
F87E 54455220
F882 54415045
F886 20534554
F88A 05
F88B 0D                         DEFB   0DH
F88C                    ;
F88C                    ; MZ-1500 MODE FLAG
F88C                    ;

F88C                    ;ORG FFFFH
F88C                    ;DEFB A5H
F88C                    ;
F88C                            SKP    H
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 51          84.06.26
FA00                       ORG    FA00H
FA00                      ;==============================
FA00                      ;
FA00                      ;   Quick-Disk(BISYNC)
FA00                      ;         Control Program
FA00                      ;
FA00                      ;      V1.0A 04.25.1984
FA00                      ;
FA00                      ;==============================
FA00                      ;
FA00                      ;
FA00                      ;
FA00                      ;
EFFF   P                  FMS:    EQU    EFFFH            ;INIT CHECK SIZE (60KByt
e)
FA00                      ;
FA00                      ;
FA00                      ;
FA00                      ;=========================
FA00                      ;
FA00                      ;         QDIO
FA00                      ;
FA00                      ;=========================
FA00                      ;
FA00                      ;
FA00                      QDIO:   ENT
FA00 3E05                         LD     A,5              ;Retry 4
FA02 324511                       LD     (RTYF),A
FA05                      ;
FA05 F3                   RTY:    DI
FA06 CD54FA                       CALL   QMEIN
FA09 FB                           EI
FA0A D0                           RET    NC
FA0B F5                           PUSH   AF
FA0C FE28                         CP     40
FA0E 2842                         JR     Z,RTY4
FA10 CDCFFC                       CALL   MTOF
FA13 3A3011                       LD     A,(QDPA)
FA16 FE04                         CP     4                ;Write ?
FA18 201D                         JR     NZ,RTY3
FA1A 3A4111                       LD     A,(FNUPF)
FA1D B7                           OR     A
FA1E 2817                         JR     Z,RTY3
FA20 AF                           XOR    A                ;FNUPF CLR
FA21 324111                       LD     (FNUPF),A
FA24 3A4211                       LD     A,(FNA)
FA27 E5                           PUSH   HL               ;RETSP <= SP-2
FA28 ED734811                     LD     (RETSP),SP
FA2C E1                           POP    HL
FA2D                      ;
FA2D F3                           DI
FA2E CD4FFC                       CALL   QDSVFN
FA31 FB                           EI
FA32 3818                         JR     C,RTY2
FA34 CDCFFC                       CALL   MTOF
FA37                      ;
FA37 F1                   RTY3:   POP    AF
FA38 F5                           PUSH   AF
FA39 FE29                         CP     41
FA3B 200F                         JR     NZ,RTY2
FA3D                      ;
```

IPLのリスト

```
** Z80 ASSEMBLER 2Z-004C <9Z-502M> PAGE 52                    84.06.26

FA3D 214511                LD      HL,RTYF
FA40 35                    DEC     (HL)
FA41 2809                  JR      Z,RTY2
FA43 F1                    POP     AF
FA44 3A4011                LD      A,(FNUPS1)
FA47 323F11                LD      (FNUPS),A
FA4A 18B9                  JR      RTY
FA4C               ;
FA4C CDC1FB        RTY2:   CALL    WRCAN
FA4F CD6DFA                CALL    QDHPC
FA52 F1            RTY4:   POP     AF
FA53 C9                    RET
FA54               ;
FA54               ;
FA54               ;
FA54 ED734811      QMEIN:  LD      (RETSP),SP
FA58 3A3011                LD      A,(QDPA)
FA5B 3D                    DEC     A
FA5C 2816                  JR      Z,QDRC          ;Ready Check
FA5E 3D                    DEC     A
FA5F 2819                  JR      Z,QDFM          ;Format
FA61 3D                    DEC     A
FA62 2860                  JR      Z,QDRD          ;Read
FA64 3D                    DEC     A
FA65 CA34FB                JP      Z,QDWR          ;Write
FA68 3D                    DEC     A
FA69 2802                  JR      Z,QDHPC         ;Head Point Clear
FA6B 1850                  JR      MTOFX           ;Motor Off
FA6D               ;
FA6D               ;=========================
FA6D               ;
FA6D               ;    Head Point Clear
FA6D               ;
FA6D               ;=========================
FA6D               ;
FA6D               QDHPC:  ENT
FA6D F5                    PUSH    AF
FA6E AF                    XOR     A
FA6F 323D11                LD      (HDPT),A
FA72 F1                    POP     AF
FA73 C9                    RET
FA74               ;
FA74               ;=========================
FA74               ;
FA74               ;    Ready Check
FA74               ;
FA74               ;=========================
FA74               ;
FA74               QDRC:   ENT
FA74 3A3111                LD      A,(QDPB)
FA77 C323FC                JP      QREDY
FA7A               ;
FA7A               ;
FA7A               ;
FA7A               ;=========================
FA7A               ;
FA7A               ;    Format
FA7A               ;
FA7A               ;=========================
FA7A               ;
```

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 53          84.06.26

FA7A                  QDFM:   ENT
FA7A AF                       XOR     A
FA7B CD4FFC                   CALL    QDSVFN
FA7E CD4DFD                   CALL    SYNCS2
FA81 01FFEF                   LD      BC,FMS
FA84 3EAA                     LD      A,AAH
FA86                  ;
FA86 2F               QDFM1:  CPL
FA87 57                       LD      D,A
FA88 CDC4FD                   CALL    TRANS
FA8B 0B                       DEC     BC
FA8C 78                       LD      A,B
FA8D B1                       OR      C
FA8E 2803                     JR      Z,QDFM2
FA90 7A                       LD      A,D
FA91 18F3                     JR      QDFM1
FA93                  ;
FA93 CD9AFD           QDFM2:  CALL    EOM
FA96 CDCFFC                   CALL    MTOF
FA99 CD82FC                   CALL    MTON
FA9C 3A4311                   LD      A,(FNB)
FA9F 3D                       DEC     A
FAA0 201E                     JR      NZ,FMERR
FAA2 CDE4FC                   CALL    SYNCL2
FAA5 01FFEF                   LD      BC,FMS
FAA8 1E55                     LD      E,55H
FAAA BB               QDFM3:  CP      E
FAAB 2013                     JR      NZ,FMERR
FAAD 0B                       DEC     BC
FAAE 78                       LD      A,B
FAAF B1                       OR      C
FAB0 2808                     JR      Z,QDFM4
FAB2 7B                       LD      A,E
FAB3 2F                       CPL
FAB4 5F                       LD      E,A
FAB5 CDDAFD                   CALL    RDATA
FAB8 18F0                     JR      QDFM3
FABA                  ;
FABA CDACFD           QDFM4:  CALL    RDCRC
FABD C3CFFC           MTOFX:  JP      MTOF
FAC0                  ;
FAC0 3E29             FMERR:  LD      A,41              ;Hard Err
FAC2 37                       SCF
FAC3 C9                       RET
FAC4                  ;
FAC4                  ;
FAC4                  ;
FAC4                  ;========================
FAC4                  ;
FAC4                  ;         Read
FAC4                  ;
FAC4                  ;========================
FAC4                  ;
FAC4                  QDRD:   ENT
FAC4 3A4411                   LD      A,(MTF)
FAC7 B7                       OR      A
FAC8 CC82FC                   CALL    Z,MTON
FACB CDFAFA                   CALL    HPS
FACE D8                       RET     C
FACF CD1FFE                   CALL    BRKC
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 54                    84.06.26

FAD2                  ;
FAD2 CDDAFD                      CALL    RDATA
FAD5 4F                          LD      C,A
FAD6 CDDAFD                      CALL    RDATA
FAD9 47                          LD      B,A
FADA 2A3411                      LD      HL,(QDPE)        ;Byte size check
FADD ED42                        SBC     HL,BC
FADF DACBFB                      JP      C,QDWE1
FAE2 2A3211                      LD      HL,(QDPC)
FAE5                  ;
FAE5                  ;Block Data Read
FAE5                  ;
FAE5 CDDAFD           BDR:       CALL    RDATA
FAE8 77                          LD      (HL),A
FAE9 23                          INC     HL
FAEA 0B                          DEC     BC
FAEB 78                          LD      A,B
FAEC B1                          OR      C
FAED 20F6                        JR      NZ,BDR
FAEF CDACFD                      CALL    RDCRC
FAF2 3A3111                      LD      A,(QDPB)
FAF5 CB47                        BIT     0,A
FAF7 20C4                        JR      NZ,MTOFX
FAF9 C9                          RET
FAFA                  ;
FAFA                  ;
FAFA                  ;Head Point Search
FAFA                  ;
FAFA                  ;
FAFA                  HPS:       ENT
FAFA 214311                      LD      HL,FNB
FAFD 35                          DEC     (HL)
FAFE 2830                        JR      Z,HPNFE          ;Not Found
FB00 CDE4FC                      CALL    SYNCL2
FB03 4F                          LD      C,A              ;BLKFLG => C reg
FB04 3A3D11                      LD      A,(HDPT)
FB07 213E11                      LD      HL,HDPTO
FB0A BE                          CP      (HL)             ;Search ok ?
FB0B 2020                        JR      NZ,HPS1
FB0D 3C                          INC     A                ;HDPT count up
FB0E 323D11                      LD      (HDPT),A
FB11 77                          LD      (HL),A           ;HDPTO count up
FB12 3A3111                      LD      A,(QDPB)
FB15 A9                          XOR     C
FB16 1F                          RRA
FB17 D0                          RET     NC               ;=
FB18                  ;
FB18                  ;Dummy read
FB18                  ;
FB18 CDDAFD           DMR:       CALL    RDATA
FB1B 4F                          LD      C,A
FB1C CDDAFD                      CALL    RDATA
FB1F 47                          LD      B,A
FB20                  ;
FB20 CDDAFD           DMR1:      CALL    RDATA
FB23 0B                          DEC     BC
FB24 78                          LD      A,B
FB25 B1                          OR      C
FB26 20F8                        JR      NZ,DMR1
FB28 CDACFD                      CALL    RDCRC
```

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 55          84.06.26

FB2B 18CD                  JR     HPS             ;next
FB2D               ;
FB2D 34            HPS1:   INC    (HL)
FB2E 18E8                  JR     DMR
FB30               ;
FB30 3E28          HPNFE:  LD     A,40            ;Not Found
FB32 37                    SCF
FB33 C9                    RET
FB34               ;
FB34               ;
FB34               ;========================
FB34               ;
FB34               ;       Write
FB34               ;
FB34               ;========================
FB34               ;
FB34               QDWR:   ENT
FB34 3A3F11                LD     A,(FNUPS)
FB37 324011                LD     (FNUPS1),A
FB3A 3A4411                LD     A,(MTF)
FB3D B7                    OR     A
FB3E 2017                  JR     NZ,QDWR1
FB40 CD82FC                CALL   MTON
FB43 3A3F11                LD     A,(FNUPS)
FB46 214311                LD     HL,FNB
FB49 86                    ADD    A,(HL)
FB4A 324311                LD     (FNB),A
FB4D 3C                    INC    A
FB4E 323D11                LD     (HDPT),A
FB51 CDFAFA                CALL   HPS
FB54 D2CBFB                JP     NC,QDWE1        ;Hard err
FB57 3A3111        QDWR1:  LD     A,(QDPB)
FB5A 47                    LD     B,A
FB5B E601                  AND    1
FB5D 200C                  JR     NZ,QDWR2
FB5F 113211                LD     DE,QDPC
FB62 78                    LD     A,B
FB63 CB97                  RES    2,A
FB65 CDD4FB                CALL   BDW
FB68 CD1FFE                CALL   BRKC
FB6B               ;
FB6B 113611        QDWR2:  LD     DE,QDPG
FB6E 3A3111                LD     A,(QDPB)
FB71 CBC7                  SET    0,A
FB73 CDD4FB                CALL   BDW
FB76 CDCFFC                CALL   MTOF
FB79 CD1FFE                CALL   BRKC
FB7C               ;
FB7C 3A3111                LD     A,(QDPB)
FB7F E604                  AND    4
FB81 284C                  JR     Z,NFNUP
FB83 3A4211                LD     A,(FNA)
FB86 213F11                LD     HL,FNUPS
FB89 86                    ADD    A,(HL)
FB8A CD4FFC                CALL   QDSVFN
FB8D 3E01                  LD     A,1
FB8F 324111                LD     (FNUPF),A
FB92 3A4211        QDWR4:  LD     A,(FNA)
FB95 214011                LD     HL,FNUPS1
FB98 86                    ADD    A,(HL)
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 56           84.06.26

FB99 3C                      INC     A
FB9A 324311                  LD      (FNB),A
FB9D 3C                      INC     A
FB9E 323D11                  LD      (HDPT),A
FBA1 CDFAFA                  CALL    HPS
FBA4 3025                    JR      NC,QDWE1
FBA6 3A3111                  LD      A,(QDPB)
FBA9 E601                    AND     1
FBAB 2007                    JR      NZ,QDWR3
FBAD 113211                  LD      DE,QDPC
FBB0 CD02FC                  CALL    BDV
FBB3 D8                      RET     C
FBB4 113611         QDWR3:   LD      DE,QDPG
FBB7 CD02FC                  CALL    BDV
FBBA D8                      RET     C
FBBB 3A4111                  LD      A,(FNUPF)
FBBE B7                      OR      A
FBBF 2807                    JR      Z,QDWR5
FBC1 AF             WRCAN:   XOR     A
FBC2 323F11                  LD      (FNUPS),A
FBC5 324111                  LD      (FNUPF),A
FBC8                ;
FBC8 C3CFFC         QDWR5:   JP      MTOF
FBCB                ;
FBCB 3E29           QDWE1:   LD      A,41
FBCD 37                      SCF
FBCE C9                      RET
FBCF                ;
FBCF CD82FC         NFNUP:   CALL    MTON
FBD2 18BE                    JR      QDWR4
FBD4                ;
FBD4                ;Block Data Write
FBD4                ;
FBD4 F5             BDW:     PUSH    AF
FBD5 213F11                  LD      HL,FNUPS
FBD8 34                      INC     (HL)
FBD9 CD4DFD                  CALL    SYNCS2
FBDC F1                      POP     AF
FBDD CDC4FD                  CALL    TRANS
FBE0 CDF8FB                  CALL    RSET
FBE3 79                      LD      A,C
FBE4 CDC4FD                  CALL    TRANS
FBE7 78                      LD      A,B
FBE8 CDC4FD                  CALL    TRANS
FBEB                ;
FBEB 7E             BDW1:    LD      A,(HL)
FBEC CDC4FD                  CALL    TRANS
FBEF 23                      INC     HL
FBF0 0B                      DEC     BC
FBF1 78                      LD      A,B
FBF2 B1                      OR      C
FBF3 20F6                    JR      NZ,BDW1
FBF5 C39AFD                  JP      EOM
FBF8                ;
FBF8                ;HL,BC SET
FBF8                ;
FBF8 EB             RSET:    EX      DE,HL
FBF9 5E                      LD      E,(HL)
FBFA 23                      INC     HL
FBFB 56                      LD      D,(HL)
```

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 57              84.06.26

FBFC 23                     INC     HL
FBFD 4E                     LD      C,(HL)
FBFE 23                     INC     HL
FBFF 46                     LD      B,(HL)
FC00 EB                     EX      DE,HL
FC01 C9                     RET
FC02                ;
FC02                ;Block Data Verify
FC02                ;
FC02 CDE4FC         BDV:    CALL    SYNCL2
FC05 CDF8FB                 CALL    RSET
FC08 CDDAFD                 CALL    RDATA
FC0B B9                     CP      C
FC0C 20BD                   JR      NZ,QDWE1
FC0E CDDAFD                 CALL    RDATA
FC11 B8                     CP      B
FC12 20B7                   JR      NZ,QDWE1
FC14                ;
FC14 CDDAFD         BDV1:   CALL    RDATA
FC17 BE                     CP      (HL)
FC18 20B1                   JR      NZ,QDWE1
FC1A 23                     INC     HL
FC1B 0B                     DEC     BC
FC1C 78                     LD      A,B
FC1D B1                     OR      C
FC1E 20F4                   JR      NZ,BDV1
FC20 C3ACFD                 JP      RDCRC
FC23                ;
FC23                        SKP     H
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 58                        84.06.26

FC23                   ;
FC23                   ;
FC23                   ;====================================
FC23                   ;    Quick-Disk
FC23                   ;        Physical iocs (bisync)
FC23                   ;
FC23                   ;        By MZ- 700 IPL-ROM
FC23                   ;           MZ-1500 IPL-ROM
FC23                   ;           MZ- 700 Disk-Basic
FC23                   ;           MZ-1500 Basic
FC23                   ;
FC23                   ;        V2.0A 04.25.1984
FC23                   ;
FC23                   ;====================================
FC23                   ;
FC23                   ;
FC23                   ; i/o port adrs
FC23                   ;
FC23                   ;SIOAD:EQU F4H; sio A data
FC23                   ;SIOBD:EQU F5H; sio B data
FC23                   ;SIOAC:EQU F6H; sio A control
FC23                   ;SIOBC:EQU F7H; sio B control
FC23                   ;
FC23                   ;
FC23                   ;
FC23                   ; Ready & Write protect
FC23                   ;    Acc = '0' : Ready check
FC23                   ;    Acc = '1' : & Write Protect
FC23                   ;
FC23                   QREDY:  ENT
FC23 47                        LD      B,A
FC24 3E02                      LD      A,02H           ;SIO hard check
FC26 D3F7                      OUT     (SIOBC),A
FC28 3E81                      LD      A,81H
FC2A D3F7                      OUT     (SIOBC),A
FC2C 3E02                      LD      A,02H
FC2E D3F7                      OUT     (SIOBC),A
FC30 DBF7                      IN      A,(SIOBC)
FC32 E681                      AND     81H
FC34 FE81                      CP      81H
FC36 C2F0FD                    JP      NZ,IOE50        ;Not ready
FC39 3E10                      LD      A,10H
FC3B D3F6                      OUT     (SIOAC),A
FC3D DBF6                      IN      A,(SIOAC)
FC3F 4F                        LD      C,A
FC40 E608                      AND     08H
FC42 CAF0FD                    JP      Z,IOE50         ;Not ready
FC45 78                        LD      A,B
FC46 B7                        OR      A
FC47 C8                        RET     Z               ;No err
FC48 79                        LD      A,C
FC49 E620                      AND     20H
FC4B C0                        RET     NZ              ;No err
FC4C C3EDFD                    JP      IOE46           ;Write protect
FC4F                   ;
FC4F                   ;
FC4F                   ; Write FN
FC4F                   ;
FC4F                   QDSVFN: ENT
FC4F F5                        PUSH    AF
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 59           84.06.26

FC50 2116FE              LD     HL,SIOSD
FC53 0609                LD     B,09H
FC55 CDBFFC              CALL   LSINT            ;save init
FC58              ;
FC58 3E10         SREDY: LD     A,10H
FC5A D3F6                OUT    (SIOAC),A
FC5C DBF6                IN     A,(SIOAC)
FC5E E608                AND    8
FC60 CAF0FD              JP     Z,IOE50          ;Not ready
FC63 3E10                LD     A,10H
FC65 D3F7                OUT    (SIOBC),A
FC67 DBF7                IN     A,(SIOBC)
FC69 E608                AND    8
FC6B 28EB                JR     Z,SREDY
FC6D 01E900              LD     BC,00E9H         ;Wait 160ms
FC70 CDFEFD              CALL   TIMW
FC73 CD82FD              CALL   SBRK             ;Send Break
FC76 CD66FD              CALL   SYNCS1           ;FN Only SYNC
FC79 F1                  POP    AF
FC7A CDC4FD              CALL   TRANS            ;FN=A
FC7D CD9AFD              CALL   EOM              ;CRC FLAG(7EH)
FC80 1833                JR     FNEND
FC82              ;
FC82              ;
FC82              ;
FC82              ; MTON -- QD MOTOR ON
FC82              ;        READ FILE NUMBER
FC82              ;        READ &CHECK CRC,FLAG
FC82              ;
FC82              MTON:  ENT
FC82 210BFE              LD     HL,SIOLD
FC85 060B                LD     B,0BH
FC87 CDBFFC              CALL   LSINT            ;load init
FC8A              ;
FC8A 3E10         LREDY: LD     A,10H
FC8C D3F6                OUT    (SIOAC),A
FC8E DBF6                IN     A,(SIOAC)
FC90 E608                AND    8
FC92 CAF0FD              JP     Z,IOE50          ;Not ready
FC95 CD1FFE              CALL   BRKC
FC98 3E10                LD     A,10H
FC9A D3F7                OUT    (SIOBC),A
FC9C DBF7                IN     A,(SIOBC)
FC9E E608                AND    8
FCA0 28E8                JR     Z,LREDY
FCA2 01E900              LD     BC,00E9H         ;Wait 160ms
FCA5 CDFEFD              CALL   TIMW
FCA8 CDFAFC              CALL   SYNCL1           ;LOAD SYNC
FCAB 324211              LD     (FNA),A
FCAE 3C                  INC    A
FCAF 324311              LD     (FNB),A
FCB2 CDACFD              CALL   RDCRC
FCB5 214711       FNEND: LD     HL,SYNCF
FCB8 CBDE                SET    3,(HL)
FCBA AF                  XOR    A
FCBB 323E11              LD     (HDPTO),A
FCBE C9                  RET
FCBF              ;
FCBF              ;
FCBF              ;      sio initial
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 60                     84.06.26

FCBF                    ;
FCBF                    ;
FCBF 0EF6               LSINT:  LD      C,F6H           ; C sioac
FCC1 EDB3                       OTIR
FCC3 3E05                       LD      A,05H
FCC5 324411                     LD      (MTF),A
FCC8 D3F7                       OUT     (SIOBC),A
FCCA 3E80                       LD      A,80H
FCCC D3F7                       OUT     (SIOBC),A
FCCE C9                         RET
FCCF                    ;
FCCF                    ; Motor off
FCCF                    ;
FCCF                    QDOFF:  ENT                     ;Basic call
FCCF                    MTOF:   ENT
FCCF F5                         PUSH    AF
FCD0 3E05                       LD      A,05H
FCD2 D3F6                       OUT     (SIOAC),A
FCD4 3E60                       LD      A,60H
FCD6 D3F6                       OUT     (SIOAC),A       ;WRGT OFF,TRANS DISABLE
FCD8 3E05                       LD      A,05H
FCDA D3F7                       OUT     (SIOBC),A
FCDC AF                         XOR     A
FCDD 324411                     LD      (MTF),A
FCE0 D3F7                       OUT     (SIOBC),A
FCE2 F1                         POP     AF
FCE3 C9                         RET
FCE4                    ;
FCE4                    ;
FCE4                    ;
FCE4                    ; SYNCL1 -- LOAD F.N SYNC ONLY
FCE4                    ;               (SEND BREAK 110ms)
FCE4                    ; SYNCL2 -- LOAD FIRST FILE SYNC
FCE4                    ;               (SEND BREAK 110ms)
FCE4                    ; SYNCL3 -- LOAD FILES SYNC
FCE4                    ;               (SEND BREAK 002ms)
FCE4                    ;
FCE4                    SYNCL2: ENT
FCE4 3E58                       LD      A,58H
FCE6 060B                       LD      B,0BH
FCE8 210BFE                     LD      HL,SIOLD
FCEB CD8BFD                     CALL    SYNCA
FCEE 214711                     LD      HL,SYNCF
FCF1 CB5E                       BIT     3,(HL)
FCF3 010300                     LD      BC,3            ;WAIT 2ms
FCF6 2805                       JR      Z,TMLPL
FCF8 CB9E                       RES     3,(HL)
FCFA 01A000             SYNCL1: LD      BC,00A0H        ;WAIT 110ms
FCFD                    ;
FCFD CDFEFD             TMLPL:  CALL    TIMW
FD00 3E05                       LD      A,05H
FD02 D3F7                       OUT     (SIOBC),A
FD04 3E82                       LD      A,82H
FD06 D3F7                       OUT     (SIOBC),A
FD08 3E03                       LD      A,03H
FD0A D3F6                       OUT     (SIOAC),A
FD0C 3ED3                       LD      A,D3H
FD0E D3F6                       OUT     (SIOAC),A
FD10 01C02C                     LD      BC,2CC0H        ;loop 220ms
FD13                    ;
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 61        84.06.26

FD13 3E10        SYNCW0: LD     A,10H
FD15 D3F6                OUT    (SIOAC),A
FD17 DBF6                IN     A,(SIOAC)
FD19 E610                AND    10H
FD1B 2808                JR     Z,SYNCW1
FD1D 0B                  DEC    BC
FD1E 78                  LD     A,B
FD1F B1                  OR     C
FD20 20F1                JR     NZ,SYNCW0
FD22 C3F6FD              JP     IOE54          ;Un format
FD25             ;
FD25 3E03        SYNCW1: LD     A,03H
FD27 D3F6                OUT    (SIOAC),A
FD29 3EC3                LD     A,C3H
FD2B D3F6                OUT    (SIOAC),A
FD2D 069F                LD     B,9FH          ;loop 3ms
FD2F             ;
FD2F 3E10        SYNCW2: LD     A,10H
FD31 D3F6                OUT    (SIOAC),A
FD33 DBF6                IN     A,(SIOAC)
FD35 E601                AND    01H
FD37 2006                JR     NZ,SYNCW3
FD39 05                  DEC    B
FD3A 20F3                JR     NZ,SYNCW2
FD3C C3F6FD              JP     IOE54          ;Un format
FD3F             ;
FD3F 3E03        SYNCW3: LD     A,03H
FD41 D3F6                OUT    (SIOAC),A
FD43 3EC9                LD     A,C9H
FD45 D3F6                OUT    (SIOAC),A
FD47 CDDAFD              CALL   RDATA
FD4A C3DAFD              JP     RDATA
FD4D             ;
FD4D             ;
FD4D             ;
FD4D             ; SYNCS1 -- SAVE F.N SYNC
FD4D             ;                (SEND BREAK 220ms)
FD4D             ; SYNCS2 -- SAVE FIRST FILE SYNC
FD4D             ;                (SEND BREAK 220ms)
FD4D             ; SYNCS3 -- SAVE FILES SYNC
FD4D             ;                (SEND BREAK 020ms)
FD4D             ;
FD4D             SYNCS2: ENT
FD4D 3E98                LD     A,98H
FD4F 0609                LD     B,09H
FD51 2116FE              LD     HL,SIOSD
FD54 CD8BFD              CALL   SYNCA
FD57 CD82FD              CALL   SBRK
FD5A             ;
FD5A 214711              LD     HL,SYNCF
FD5D CB5E                BIT    3,(HL)
FD5F 011D00              LD     BC,001DH       ;WAIT 20ms
FD62 2805                JR     Z,TMLPS
FD64 CB9E                RES    3,(HL)
FD66 014001      SYNCS1: LD     BC,0140H       ;WAIT 220ms
FD69             ;
FD69 CDFEFD      TMLPS:  CALL   TIMW
FD6C 3E05                LD     A,05H
FD6E D3F6                OUT    (SIOAC),A
FD70 3EEF                LD     A,EFH
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 62                    84.06.26

FD72 D3F6                    OUT     (SIOAC),A
FD74 010100                  LD      BC,1             ;WAIT 0.7ms
FD77 CDFEFD                  CALL    TIMW
FD7A 3EC0                    LD      A,C0H
FD7C D3F6                    OUT     (SIOAC),A
FD7E 3EA5                    LD      A,A5H
FD80 1842                    JR      TRANS
FD82               ;
FD82               ;
FD82               ;
FD82               ; SBRK -- SEND BREAK (00H)
FD82               ;
FD82               SBRK:     ENT
FD82 3E05                    LD      A,05H
FD84 D3F6                    OUT     (SIOAC),A
FD86 3EFF                    LD      A,FFH
FD88 D3F6                    OUT     (SIOAC),A
FD8A C9                      RET
FD8B               ;
FD8B               ;
FD8B               ;
FD8B 0EF6          SYNCA:    LD      C,F6H            ; C sioac
FD8D ED79                    OUT     (C),A
FD8F 3E05                    LD      A,5
FD91 D3F7                    OUT     (SIOBC),A
FD93 3E80                    LD      A,80H
FD95 D3F7                    OUT     (SIOBC),A
FD97 EDB3                    OTIR
FD99 C9                      RET
FD9A               ;
FD9A               ;
FD9A               ;
FD9A               ;
FD9A               ; EOM -- End off message
FD9A               ;          Save CRC#1,#2,FLAG
FD9A               ;          File space check
FD9A               ;
FD9A               ;
FD9A               EOM:      ENT
FD9A 010100                  LD      BC,1             ;WAIT 0.7ms
FD9D CDFEFD                  CALL    TIMW
FDA0 3E10                    LD      A,10H
FDA2 D3F7                    OUT     (SIOBC),A
FDA4 DBF7                    IN      A,(SIOBC)
FDA6 E608                    AND     8
FDA8 C0                      RET     NZ
FDA9 C3F3FD                  JP      IOE53            ;NO file space
FDAC               ;
FDAC               ; RDCRC -- READ CRC & CHECK
FDAC               ;
FDAC               RDCRC:    ENT
FDAC 0603                    LD      B,3
FDAE CDDAFD        RDCR1:    CALL    RDATA
FDB1 10FB                    DJNZ    RDCR1
FDB3 DBF6          RDCR2:    IN      A,(SIOAC)
FDB5 0F                      RRCA
FDB6 30FB                    JR      NC,RDCR2         ; Rx Available
FDB8 3E01                    LD      A,01H
FDBA D3F6                    OUT     (SIOAC),A
FDBC DBF6                    IN      A,(SIOAC)
```

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 63          84.06.26

FDBE E640                 AND   40H
FDC0 202B                 JR    NZ,IOE41        ;Hard err
FDC2 B7                   OR    A
FDC3 C9                   RET
FDC4             ;
FDC4             ;
FDC4             ; Save 1 chr by Acc
FDC4             ;         & ready check
FDC4             ;
FDC4             TRANS:   ENT
FDC4 F5                   PUSH  AF
FDC5 DBF6        TRA1:    IN    A,(SIOAC)
FDC7 E604                 AND   4               ;TRANS buf null
FDC9 28FA                 JR    Z,TRA1
FDCB F1                   POP   AF
FDCC D3F4                 OUT   (SIOAD),A
FDCE 3E10        NRCK:    LD    A,10H
FDD0 D3F6                 OUT   (SIOAC),A
FDD2 DBF6                 IN    A,(SIOAC)
FDD4 E608                 AND   08H
FDD6 CAF0FD               JP    Z,IOE50         ;Not ready
FDD9 C9                   RET
FDDA             ;
FDDA             ; Read data (1 chr)
FDDA             ;
FDDA             RDATA:   ENT
FDDA CDCEFD               CALL  NRCK
FDDD DBF6                 IN    A,(SIOAC)       ;RR0
FDDF 07                   RLCA
FDE0 3808                 JR    C,IOE41         ;Hard err
FDE2 0F                   RRCA
FDE3 0F                   RRCA
FDE4 30F4                 JR    NC,RDATA
FDE6 DBF4                 IN    A,(SIOAD)
FDE8 B7                   OR    A
FDE9 C9                   RET
FDEA             ;
FDEA             ;
FDEA             ; i/o err
FDEA             ;
FDEA 3E29        IOE41:   LD    A,41            ;Hard err
FDEC 21                   DEFB  21H
FDED 3E2E        IOE46:   LD    A,46            ;Write protect
FDEF 21                   DEFB  21H
FDF0 3E32        IOE50:   LD    A,50            ;Not ready
FDF2 21                   DEFB  21H
FDF3 3E35        IOE53:   LD    A,53            ;No file space
FDF5 21                   DEFB  21H
FDF6 3E36        IOE54:   LD    A,54            ;Un format
FDF8 ED7B4811             LD    SP,(RETSP)
FDFC 37                   SCF
FDFD C9                   RET
FDFE             ;
FDFE             ;
FDFE             ;=================================
FDFE             ;
FDFE             ;     wait  timer
FDFE             ;
FDFE             ;  BC=001H=  0.7ms(  0.704ms)
FDFE             ;     003H=  2.0ms(  2.107ms)
```

```
FDFE                ;       01DH= 20.0ms( 19.938ms)
FDFE                ;       0A0H=110.0ms(110.050ms)
FDFE                ;       0E9H=160.0ms(160.140ms)
FDFE                ;       140H=220.0ms(219.940ms)
FDFE                ;
FDFE                ;==================================
FDFE                ;
FDFE F5             TIMW:   PUSH    AF
FDFF 3E98           TIMW1:  LD      A,98H
FE01 3D             TIMW2:  DEC     A
FE02 20FD                   JR      NZ,TIMW2
FE04 0B                     DEC     BC
FE05 78                     LD      A,B
FE06 B1                     OR      C
FE07 20F6                   JR      NZ,TIMW1
FE09 F1                     POP     AF
FE0A C9                     RET
FE0B                ;
FE0B                ;
FE0B                ;=================================
FE0B                ;
FE0B                ;    SIO CH A COMMAND CHAIN
FE0B                ;
FE0B                ;    SIOLD -- LOAD INIT. DATA
FE0B                ;    SIOSD -- SAVE INIT. DATA
FE0B                ;
FE0B                ;=================================
FE0B                ;
FE0B 58             SIOLD:  DEFB    58H             ;CHANNEL RESET
FE0C 04                     DEFB    04H             ;POINT WR4
FE0D 10                     DEFB    10H             ;X1 CLOCK
FE0E 05                     DEFB    05H             ;POINT WR1
FE0F 04                     DEFB    04H             ;CRC-16
FE10 03                     DEFB    03H             ;POINT WR3
FE11 D0                     DEFB    D0H             ;ENTER HUNT PHASE
FE12                ;Rx 8bits
FE12 06                     DEFB    06H             ;POINT WR6
FE13 16                     DEFB    16H             ;SYNC CHR(1)
FE14 07                     DEFB    07H             ;POINT WR7
FE15 16                     DEFB    16H             ;SYNC CHR(2)
FE16                ;
FE16 98             SIOSD:  DEFB    98H             ;CHANNEL RESET
FE17                ;Tx CRC Generator reset
FE17 04                     DEFB    04H             ;POINT WR4
FE18 10                     DEFB    10H             ;X1 CLOCK
FE19 06                     DEFB    06H             ;POINT WR6
FE1A 16                     DEFB    16H             ;SYNC CHR(1)
FE1B 07                     DEFB    07H             ;POINT WR7
FE1C 16                     DEFB    16H             ;SYNC CHR(2)
FE1D 05                     DEFB    05H             ;POINT WR5
FE1E 6D                     DEFB    6DH             ;Tx CRC ENABLE
FE1F                ;
FE1F                ;
FE1F                ; BREAK CHECK
FE1F                ;
FE1F 3EE8           BRKC:   LD      A,E8H
FE21 3200E0                 LD      (E000H),A
FE24 00                     NOP
FE25 3A01E0                 LD      A,(E001H)
FE28 E681                   AND     81H
```

```
**  Z80 ASSEMBLER 2Z-004C <9Z-502M>  PAGE 65          84.06.26

FE2A C0                  RET    NZ
FE2B CDC1FB              CALL   WRCAN
FE2E ED7B4811            LD     SP,(RETSP)
FE32 37                  SCF
FE33 C9                  RET
FE34                 ;
FE34                     END
```

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 66          84.06.26

#CLR8   09DE   ..LPTX EBCF   .4DE   02A6   .LPTX  EBCA   12ERMG EE9A
12NSET  F854   12RET  F781   1R12RO F734   1R12RD F72D   1R12WR F7BE
2HEX    041F   ?ADCN  0BB9   ?BEL   0577   ?BLNK  0DA6   ?BRK   0A32
?CLER   0FD8   ?DPCT  0DDC   ?ERX   EB4C   ?ERX0  EB56   ?KEY   08CA
?MODE   073E   ?PRTS  0920   ?RDD   04F8   ?RDI   04D8   ?TMST  0308
?VRFY   0588   ?WRD   0475   ?WRI   0436   ADJ    F69D   AINPO  ED69
ALEXCD  002A   ASC    03DA   ATRB   10F0   ATRCO  F280   BDR    FAE5
BDSKCD  0039   BDV    FC02   BDV1   FC14   BDW    FBD4   BDW1   FBEB
BELL    003E   BFRF   1146   BGETL  0126   BGETLX EB5C   BOOT   F532
BPARA   CEE9   BPRD   CF00   BREAD  F608   BRKC   FE1F   BRKCD  0000
BRKCK   001E   BRKEY  001E   BSDCD  0003   BSYO   F602   BSYOFO F5F0
BSYOF1  F5FD   BSYOF2 F5ED   BSYOF3 F5F8   BSYOFE F600   BSYOFF F5E8
BSYON   F5C9   BSYONO F5D1   BSYON1 F5DE   BSYON2 F5CE   BSYON3 F5D9
BSYONE  F5E1   BTXCD  0002   BUF    11A3   BUFER  11A3   BUFF   11A3
CHECK   F7A2   CK1    F7A9   CK2    F7AE   CLBFO  CEF6   CLBF1  CEF7
CLBF2   CEF8   CLBF3  CEF9   CLLET  EA8A   CMD    CEF4   CMDOT1 F5B6
CMDOT2  F6AF   CMYO   005B   CMYOX  ECA2   CNVRT  F6F7   COMNT  1108
CR      00D8   CRMSG  F85E   CRMSGC F857   CSS    E974   DIREFC F0BE
DIRIOP  F09E   DIRMSG F46C   DIRMTF F0C6   DISCLR EAA1   DLY1M  F71B
DLY60M  F722   DLY80U F714   DLYT   F726   DM     00DC   DMR    FB18
DMR1    FB20   DR     00DB   DRDISO F117   DRDIS1 F114   DSFLNA EF24
DSILF   E9A1   DSPATR F0DA   DSPXY  1171   DTADR  1104   DUM1X  ED10
DUM2X   ECC6   DUM3X  ECBF   DUMPX  ECB1   EAINP  ED61   EDFLCK F050
EOM     FD9A   ERCK1  F6E4   ERJMP  F6F1   ERRET  F6EF   ERRET1 F6EB
ERRMO   EE3B   ERRM1  EE4E   ERRM2  EE5E   ERRM3  EE72   ERRM4  EE85
ERRMG   EA7B   ERROR  F6C2   ERRTRO F520   ERRTR1 F523   ERRTRT F517
EXADR   1106   EXCLDF E9ED   EXLDF  EF66   EXLDFL E9DF   FDB    E90F
FDCC&   E918   FDCK   EB39   FDEXC  F508   FDMTON F578   FDX    F47F
FDX1    F4C6   FDX2   F4CC   FDX3   F4D1   FEDSCH F037   FILESH EF18
FILSCH  EF85   FMERR  FAC0   FMS    EFFF   FNA    1142   FNB    1143
FNEND   FCB5   FNINP  ED16   FNINPO ED2D   FNINP1 ED3A   FNINP2 ED42
FNINP3  ED46   FNINP4 ED4E   FNINP5 ED58   FNINP6 ED5A   FNINPR ED2B
FNUPF   1141   FNUPS  113F   FNUPS1 1140   GETKY  001B   GETL   0003
GOTO    00EA   GOTOX  EC4D   HDERCD 0029   HDPCL  F318   HDPT   113D
HDPTO   113E   HEXIYX EB6A   HLHEX  0410   HPNFE  FB30   HPS    FAFA
HPS1    FB2D   HS     00DD   INTER  F6B9   IOE41  FDEA   IOE46  FDED
IOE50   FDF0   IOE53  FDF3   IOE54  FDF6   IOFRS  EF7A   IPLMO  EDD4
IPLM1   EDE6   IPLM2  EDF7   IPLM3  EE07   IPLM4  EE17   IPLMC  F52B
JST1X   EB1C   KEYCK  E8F8   KEYPA  E000   KEYPB  E001   KSCANO E948
LDFNCK  EFA9   LEACO  EF34   LET    EA8F   LETNL  0006   LLPT   0470
LOAOX   EB79   LOADX  EB74   LPARAO EF40   LPARA1 EF43   LPRNT  EBE3
LREDY   FC8A   LSINT  FCBF   MASTE  F50F   MCHECK F4A6   MCORX  EC1A
MCR1X   EC1D   MCR2X  EC46   MCR3X  EC49   MGALE  F3C2   MGBDE  F421
MGBRK   F433   MGHDE  F3B4   MGNFE  F388   MGNRE  F3E9   MGNSE  F3F8
MGTME   F39C   MGUFE  F40F   MGWPT  F3D6   MLDSP  02BE   MOFF   F591
MON     E96A   MONMSG EEAF   MSG    0015   MSG?3  06E7   MSGCR  F85A
MSGE1   0147   MSGEA  EEE1   MSGLD  EEFB   MSGLET EA95   MSGOK  0942
MSGPS   EA97   MSGQ01 F325   MSGQ02 F330   MSGQ03 F33B   MSGQ04 F346
MSGQ05  F351   MSGQ07 F35C   MSGQ10 F367   MSGQ11 F372   MSGQ?? F37D
MSGQOK  F321   MSGSVO EEC7   MSGTA  EED4   MSGW01 F863   MSGW02 F87A
```

IPLのリスト

```
MSGXA  EEEE  MTD1   F57E  MTF    1144  MTFG   CEF5  MTOF   FCCF
MTOFX  FABD  MTON   FC82  NAME   10F1  NAMSIZ 0011  NFNUP  FBCF
NFSECD 0035  NKYWT  F220  NKYWTO F222  NL     0009  NLPHL  05FA
NOTIO  F51B  NRCK   FDCE  NTFECD 0028  OBJCD  0001  PALLET E886
PALRES E80A  PALT   E881  PARST1 F65C  PARST2 F68C  PCGC   E89B
PCGCL  E89F  PCGCLO E8A4  PCGCLR E8AB  PCGRES E88D  PENX   EBDF
PIOA   E8DE  PIOB   E8CF  PIOINI E8E0  PLOTX  EBD8  PMSG   EBF9
PMSG1  EBFC  PRNT   0012  PRNT3  096C  PRNTS  000C  PRTHL  03BA
PSGRES E82A  PSGRS  E82F  PSGRSO E807  PSGSET E833  PTCJST EBC7
PTESTX EBA9  PTRNX  EBD4  PTSTOX EBAE  PTST1X EBC4  QBFLSH EA30
```

IPLのリスト

```
**  Z80 ASSEMBLER 2Z-004C  <9Z-502M>  PAGE 67              84.06.26

QBT    EA0E  QC     F1B8  QCSPST F1FB  QD     F087 .QDATRC EF1D
QDCPA  113A  QDCPB  113B  QDCPC  113C  QDFM   FA7A  QDFM1  FA86
QDFM2  FA93  QDFM3  FAAA  QDFM4  FABA  QDFMER EA4A  QDFMG  F43C
QDHPC  FA6D  QDIO   FA00  QDIOS  E80D  QDIRBF CD90  QDLDER EA54
QDNTR  EA47  QDOFF  FCCF  QDOKM  F02D  QDOKMO F13B  QDPA   1130
QDPB   1131  QDPC   1132  QDPE   1134  QDPG   1136  QDPI   1138
QDQCM  F44C  QDQCMO F463  QDRC   FA74  QDRCK  EFB5  QDRD   FAC4
QDRWCK F079  QDSVFN FC4F  QDTADR 1106  QDTBL  1130  QDWE1  FBCB
QDWR   FB34  QDWR1  FB57  QDWR2  FB6B  QDWR3  FBB4  QDWR4  FB92
QDWR5  FBC8  QER00  EF55  QER02  F007  QER03  F02A  QER04  F0C3
QER05  F155  QER06  F1EF  QER07  F212  QER08  F2AF  QER1   F2C9
QER2   F2D0  QER3   F2D7  QER4   F2DE  QER5   F2E5  QER6   F2EC
QER7   F2F3  QER8   F2FA  QERMF  F2FD  QERTO  F2C2  QERTRT F2BB
QEXADR 1108  QF     F13E  QL     EF06  QLINF  EF97  QMEIN  FA54
QNTRCD 0032  QREDY  FC23  QS     EFC1  QSIOST F009  QSIZE  1104
QSVIOS F00C  QUICK  EAF7  QX     F231  QXFNO  F257  QXSPST F284
QXST1  F24E  R12B0  E933  R12B1  E939  R12C&B E924  R12CK1 F82E
RAMBRD EB1E  RCLB   F5A9  RDA    EC0A  RDATA  FDDA  RDCR1  FDAE
RDCR2  FDB3  RDCRC  FDAC  RDDAT  002A  RDDTBL EBA1  RDINF  0027
RDTB0  E9D9  RE3    F641  RE4    F622  RE6    F620  RE8    F60E
READ1  F60B  READY  F53D  REDY0  F555  REDY1  F573  REDY2  F572
REND   F657  RETSP  1148  RMBD1  EB2C  RSET   FBF8  RTY    FA05
RTY2   FA4C  RTY3   FA37  RTY4   FA52  RTYF   1145  SAVEX  EC51
SBRK   FD82  SCR    00DA  SD&VC  E83A  SEEK   F589  SELM0  ED85
SELM1  ED97  SELM2  EDA7  SELM3  EDB6  SELM4  EDC8  SELMG0 E942
SELMSG EA56  SFNC   F069  SFNC0  F075  SGX    ECA5  SIDST  F67C
SIOAC  00F6  SIOAD  00F4  SIOBC  00F7  SIOBD  00F5  SIOLD  FE0B
SIOSD  FE16  SIZE   1102  SJP    F497  SLMSG  E93F  SLPT   03D5
SP     10F0  SPHEX  03B1  SRCK0  F1A1  SRCK1  F1AC  SRCK2  F1A7
SREDY  FC58  SSX    E871  STOX   E843  ST1X   EAA7  ST2X   EAB5
STAFG  CEFB  STARTO E800  STARTX E810  STRT00 E804  SUM    F782
SUMCK1 F78B  SUMCK2 F79E  SUMCK3 F792  SVFLST F002  SWRK   119D
SYNCA  FD8B  SYNCF  1147  SYNCL1 FCFA  SYNCL2 FCE4  SYNCS1 FD66
SYNCS2 FD4D  SYNCW0 FD13  SYNCW1 FD25  SYNCW2 FD2F  SYNCW3 FD3F
TAINP  ED5C  TEMPW  119E  TIMIN  038D  TIMW   FDFE  TIMW1  FDFF
TIMW2  FE01  TMFECD 0033  TMLPL  FCFD  TMLPS  FD69  TPERJP EA0B
TPERX  EA01  TPERX0 EA03  TR     00D9  TRA1   FDC5  TRANS  FDC4
TRANS0 F703  TRANS1 F70A  UNFMCD 0036  VRFCNT CEFA  VRFYX  EC95
WAITY  F15A  WATIO  F17A  WATI1  F199  WATIM  F177  WATKO  F169
WATK1  F174  WATKY  F164  WCK1   F80A  WCK2   F80E  WCK3   F812
WMOD0  E977  WMOD1  E997  WPRTCD 002E  WRCAN  FBC1  XAINP  ED66
```

IPLのリスト

3-1-3 モニタについて

BASICにおいておもな内部機能は，モニタにその入口をもっており，機械語プログラムでこれらの機能を利用するときには，これを使用できるようになっています．

モニタ機能の利用方法は，次の2種類があります．

① モニタサブルーチン

大部分のモニタサブルーチンは次の方法で実行します．

○ Ent regに必要な情報を格納（必要ないことも多い）

⇩

○先頭アドレスをCALLすると，その機能が実行される

⇩

○リターン情報がある場合はExt regに格納されている

例：キーボードから1文字入力する場合，ただしキーが押されるまで待つ(INKEY$の項目参照)．

```
LD A, 1
CALL  58 H
```

以上，このプログラムを実行すると，Aレジスタにキーのコードが返ってくるので，これ以降で利用することになります．

例外的にcalling sequenceを記載してある場合はこの方法で使用します．

例：Aregが，1，2，3のいずれかと一致するかチェックする(CHKACCの項目参照)．

```
CALL  11 BH
DEFB  3
DEFB  1
DEFB  2
DEFB  3
```

以上このプログラムを実行すると，どれかに一致していればゼロフラグがsetされます．

② コーリングシーケンス

コーリングシーケンスとして登録してある機能は次の方法で実行します．

○ Ent regに必要な情報を格納（必要ないことも多い）

⇩

○次の手順でこの機能を実行する．

```
RST  18 H
DEFB  n
```

ここでnはコールNo.です．

例：プリンタへ，コード変換せずに"A"を出力する．(LPTOTの項目参照)

```
LD  A, 41 H
RST  18 H
DEFB  06 H
```

以上でOKです．

ただしファイル関係やMUSIC関係など，コールする順番があるので，注意しないといけません．

以下にその機能別の説明を掲載しますが，その見方についてを説明しましょう．

Function……機能の概略説明である

Ent reg……実行に先だって必要な情報と，それを格納するレジスタ

Ext reg……実行後，結果，または状態が格納されてくるレジスタ

Rsv reg……実行しても内容が変わらないレジスタ

タイトル行の例

GETL　0003H
→先頭アドレス
→項目名

0 0　MONOP
→項目名
→コールNO.

項目名は分類するためにつけた名称であり，使用するときは特に必要ではありません．

name	adrs	leng	function
INPFLG	004d	1	0 : PLOT OFF, 1 : PLOT ON
SCRLFG	000e	1	if =0, no scroll window, roll up/down ok. (700 only)
LINLIM	004f	1	max length of 1 line
CMTMSG	0050	1	if =0, display "↓PLAY", "WRITING...", etc.
CURX	0054	1	cursor X
CURY	0055	1	cursor Y
CURYST	0056	1	Y scroll start
CURYED	0057	1	Y scroll end
CURXST	005b	1	X scroll start
CURXED	005c	1	X scroll end
COLORF	005d	1	CCOLOR
CROMSL	005e	1	CG·ROM select
CURMAK	0060	1	cursor mark (disp-code)
CONTTB	0067	32	control code jump table 16 * 2byte
SCRNTO	00a7	51	line concatenate flag
SYSSTA	00e3	1	hot start jump adrs
ERRORP	0188	1	error handling jump adrs
ELMD	1000	1	(work area for directry) file mode
ELMD1	1001	17	〃 file name + 0DH
ELMD18	1012	1	〃 lock
ELMD19	1013	1	〃 system id
ELMD20	1014	2	〃 size
ELMD22	1016	2	〃 loading adrs
ELMD24	1018	2	〃 execute adrs
ELMD26	101a	6	〃 reserved
ELMD32	1020	32	〃 reserved
ZTOP	1040	2	pointer for lu-number table
ZLOG	1042	1	(lu-work area) lu-number
ZRWX	1043	1	〃 1 : ropen, 2 : wopen, 4 : xopen
ZEQT	1044	2	〃 adrs of device-table
ZCH	1046	1	〃 channel
ZEOF	1047	1	〃 0 : not eof, 80H : eof
ZWRK1	1048	1	〃 work area 1
ZWRK2	1049	1	〃 work area 2
ZNXT	104a	2	(device-table area) next device chain
ZDEVNM	104c	4	〃 device name + 00H
....	1050	23	〃 work
DCHAN	1069	1	default channel no.
DDEV	106a	2	default device adrs

モニタワークエリア

name	adrs	leng	function
TEXTST	1070	2	BASIC text area start
POOL	1072	2	IOCS work area start
VARST	1074	2	BASIC variable area start
STRST	1076	2	BASIC string area start
VARED	1078	2	BASIC temporary area start
TMPEND	107a	2	BASIC temporary area end
INTFAC	107c	2	BASIC floating-acc area start
MEMLMT	107e	2	BASIC LIMIT instruction
MEMMAX	1080	2	last available memory
POINTX	1088	2	(graphic work) position-X
POINTY	108a	2	〃 position-Y
CRTMOD	108c	1	〃 0 : G-mode, 1 : I-mode
COL	108d	1	〃 default color
BITFU	108e	1	〃 default write mode
PRTYD	108f	1	〃 priority data
DISPX	1090	1	(CRT/LPT work) 0 : MSG, 1 : MSGX
FILOUT	1091	1	〃 0 : CRT, 1 : LPT (for &CR, &1C, &1CX, &MSG)
PSEL	1092	1	〃 printer select
PCRLF	1093	1	〃 crlf code (0AH or 0DH)
·········	1094	3	〃 work
PNMODE	1097	1	〃 plotter mode, 1 : textr, 2 : graph
·········	1098	8	〃 reserved
·········	10a0	90	monitor stack area
IBUFE	10f0	128	CMT work
·········	1170	39	reserved
·········	1197	4	CMT work
·········	119b	3	reserved
TEMPW	119e	1	tempo data
·········	119f	4	music work
KEYBM1	11a3	1	work
KEYBUF	11a4	270	key input buffer
·········	12b2	14	reserved
FUNBUF	12c0	160	function key define
INBUFC	1360	1	pending key counter
INBUFL	1361	1	pending key length
INBUF	1362	16	pending key buffer

● adrs is hexadecimal, length is decimal.

モニタワークエリア

GETL　　　0003H
Function　KB から 1 行入力. データの最後には 00 H が入る.
Ent reg. DE buffer adrs
Ext reg. if BREAK then CF:= 1
Rsv reg. except AF

CR1　　　0006H
Function CRT を改行する.
Ent reg. non
Ext reg. non
Rsv reg. all

CR2　　　0009H
Function　cursor が行の先頭でなければ改行.
Ent reg. non
Ext reg. non
Rsv reg. all

CRT1S　　　000CH
Function　CRT へスペースを 1 文字出力.
Ent reg. non
Ext reg. non
Rsv reg. except AF

PRNTT　　　000FH
Function　CRT へ 10 文字単位のタブを出力.
Ent reg. non
Ext reg. non
Rsv reg. except AF

CRT1C　　　0012H
Function　CRT へ 1 文字出力. 制御コードは実行する.
Ent reg. A ascii code
Ext reg. non
Rsv reg. all

CRTMSG　　　0012H
Function　CRT へ文字列表示. エンドコードは 00 H. 制御コードは実行する.
Ent reg. DE 文字列のポインタ.
Ext reg. non
Rsv reg. all

INKEY 0　　　001BH
Function　KB をスキャンし, ASCII コード 1 文字入力. 押されていなければ, 00 H.
Ent reg. non
Ext reg. A mz-ascii-code
Rsv reg. except AF

BRKCHK　　　001EH
Function　BREAK キーをスキャンする.
Ent reg. non
Ext reg. if break then ZF:= 1
Rsv reg. except AF

SAVE1　　　0021H
Function　information block を CMT に SAVE する.
Ent reg. HL top adrs of information-block.
　　BC byte-size of information-block.
　　(normally 128)
Ext reg. if break then CF:= 1
Rsv reg. non

SAVE2　　　0024H
Function　data block を CMT に SAVE する.
Ent reg. HL top adrs of data-block.
　　BC byte-size of data-block.
Ext reg. if break then CF:= 1
Rsv reg. non

LOAD1　　　0027H
Function　information-block を CMT より LOAD する.
Ent reg. HL top adrs of information-block
　　BC byte-size of information-block
Ext reg. if break then CF:= 1, A<>2.
　　if checksum err then CF:= 1, A= 2.
Rsv reg. non

LOAD2　　　002AH
Function data block を CMT より LOAD する.
Ent reg. HL top adrs of data-block.
　　BC byte-size of data-block.
Ext reg. if break then CF:= 1, A<>2.
　　if checksum err then CF:= 1, A= 2.
Rsv reg. non

LOAD?2　　002DH
Function　data block の VERIFY.
Ent reg. HL top adrs of data-block.
　BC byte-size of data-block.
Ext reg. if break then CF:＝1，A< > 2.
　if error then CF:＝1，A＝2.
Rsv reg. non

MUSIC　　0030H
Function　音楽を演奏する.
Ent reg. DE music data pointer. end-code is 0DH.
Ext reg. non
Rsv reg. non

TIMST　　0033H
Function　time set
Ent reg. DE time (second)
　A　AM／PM……AM のとき 0，PM のとき 1
Ext reg. non
Rsv reg. except AF

TIMRD　　003BH
Function　time read
Ent reg. non
Ext reg. DE time (second)
　A　AM／PM……AM のとき 0，PM のとき 1
Rsv reg. except AF, DE

BEEP　　003EH
Function　BEEP 音を鳴らす.
Ent reg. non
Ext reg. non
Rsv reg. except AF.

TEMPO　　0041H
Function　tempo set
Ent reg. A　tempo 01H…07H
Ext reg. non
Rsv reg. except AF

MLDSP　　0047H
Function music stop
Ent reg. non
Ext reg. non
Rsv reg. except AF

INKEY$　　0058H
Function　KB から 1 文字入力.
Ent reg. A＝0 キーをスキャンし，押されていれば 1 文字入力.
　A＝1 キーが押されるまで待つ.
　A＝FFH キーをスキャンし，押されていれば 1 文字入力するが，以前に入力した文字と同じときは入力しない.
Ext reg. A mz-ascii-code
Rsv reg. except AF

LDAHL　　00E6H
Function　VRAM／IO／control バンクのデータの読出し.
Ent reg. HL adrs
Ext reg. A data
Rsv reg. except A

LDHLA　　00EEH
Function　VRAM／IO／control バンクへデータを書込む.
Ent reg. HL adrs
　A data
Ext reg. non
Rsv reg. non

PUSHR　　0108H
Function subroutine で HL, DE, BC, IX レジスタをセーブする.
Ent reg. non
Ext reg. non
Rsv reg. except IX

CHKACC　　011BH
Function　A-reg をいくつかのデータと比較
Calling CALL CHKACC
sequence DEFB number-of-data
　DEFB data1
　DEFB data2
　:　:
　DEFB data n

Ent reg. A 比較データ
Ext reg. ZF
Rsv reg. except flags

LDDEMI　012DH
Function DE:=(HL); HL:=HL+2;

LDDEMD　0132H
Function DE:=(HL);

?CLRHL　0137H
Function メモリの一定領域を0にする(HL):=0; HL:=HL+1; B:=B−1;
until B=0;

?SETHL　0138H
Function メモリの一定領域をA-regの値にする(HL):=A; HL:=HL+1; B:=B−1;
until B=0;

?CLRDE　013DH
Function メモリの一定領域を0にする(DE):=0; DE:=DE+1; B:=B−1;
until B=0;

?SETDE　013EH
Function メモリの一定領域をA-regの値にする(DE):=A; DE:=DE+1; B:=B−1;
until B=0;

LDDEHL　0143H
Function ブロック転送(DE):=(HL); DE:=DE+1; HL:=HL+1; B:=B−1;
until B=0;

LDHLDE　014AH
Function ブロック転送(HL):=(DE); DE:=DE+1; HL:=HL+1; B:=B−1;
until B=0;

INDRCT　0151H
Function HL:=(HL);

ADDHLA　0158H
Function HL:=HL+A;

HLFTCH　015EH
Function while (HL)=" " do HL:=HL+1;
A:=(HL);

TEST1　0138H
Calling sequence CALL TEST1
DEFB keyword
Function call HLFTCH;
if A=keyword then HL:=HL−1;
Ent reg. HL data pointer
Ext reg. A next character
ZF A=keyword ?

モニタサブルーチン

00 .MONOP
Function RAM monitor
Ent reg. non
Ext reg. non
Rsv reg. non

01 .CR1
Function CRTを改行する．
Ent reg. non
Ext reg. non
Rsv reg. all

02 .CR2
Function cursorが行の先頭でなければ改行する．
Ent reg. non
Ext reg. non
Rsv reg. all

03 .CRT1C
Function CRTへ1文字出力．制御コードは実行する．
Ent reg. A ascii code
Ext reg. non
Rsv reg. all

04 .CRT1X

モニタコーリングシーケンス

Function CRT へ1文字出力．制御コードはリバース表示．
Ent reg. A ascii code
Ext reg. non
Rsv reg. all

05 .CRTMS
Function CRT へ文字列表示．エンド・コードは 00 H. CRT1C 相当．
Ent reg. DE 文字列のポインタ．
Ext reg. non
Rsv reg. all

06 .LPTOT
Function プリンタへ1文字出力. コード変換なし.
Ent reg. A データ
Ext reg. non
Rsv reg. all

07 .LPT1C
Function プリンタへ1文字出力. 各プリンタの種類に合わせてコード変換する．
Ent reg. A データ
Ext reg. non
Rsv reg. all

08 .&CR
Function CRT かプリンタを改行する．FILOUT (アドレス 1091 H) の値が0のとき CRT, 1のときプリンタを選択する．
Ent reg. A データ
Ext reg. non
Rsv reg. all

09 .&1C
Function CRT かプリンタへ1文字出力. CRT1C 相当．
Ent reg. A データ
Ext reg. non
Rsv reg. all

0A .&1CX
Function CRT かプリンタへ1文字出力. CRT1X 相当．
Ent reg. A データ
Ext reg. non
Rsv reg. all

0B .&MSG
Function CRT かプリンタへ文字列表示. エンド・コードは 00 H. CRT1C 相当.
Ent reg. DE 文字列のポインタ.
Ext reg. non
Rsv reg. all

0C .GETL
Function KB から1行入力. データの最後に 00 H が入る.
Ent reg. DE buffer adrs
Ext reg. if BREAK then CF:=1
Rsv reg. except AF

0D .INKEY
Function KB から1文字入力.
Ent reg. A=0 キーをスキャンして押されていればそのキーをとり込む.
A=1 キーが押されるまで待つ.
A=FFH キーをスキャンして押されていればそのキーをとり込むが，以前のキーと同じときはとり込まない.
Ext reg. A mz-ascii-code
Rsv reg. except AF

0E .BREAK
Function BREAK キーをスキャンする.
Ent reg. non
Ext reg. if break then ZF:=1
Rsv reg. except AF

0F .HALT
Function SPACE key が押されていたら, もう1度押されるまで待つ.
BREAK key が押されたら, JP BREAKZ
Ent reg. non
Ext reg. non
Rsv reg. except AF

10 .DI
Function music wait, spool stop and disable

interrupt.
Ent reg. non
Ext reg. non
Rsv reg. except AF

11 .EI
Function spool start and enable interrupt
Ent reg. non
Ext reg. non
Rsv reg. except AF

13 .DEASC
Function convert ascii-numeric to binary
Ent reg. HL pointer
Ext reg. DE binary data
HL point next adrs
Rsv reg. except AF, HL, DE

14 .DEHEX
Function convert ascii-hex to binary
Ent reg. HL pointer
Ext reg. DE binary data
HL point next adrs
Rsv reg. except AF, HL, DE

15 .CKHEX
Function check hex
Ent reg. A ascii code
Ext reg. if hex then CF:=0, A:=binary
else CF:=1
Rsv reg. except AF

16 .ASCHL
Function convert binary to ascii-numeric
Ent reg. HL binary data
DE buffer adrs
if B=0 then zero-suppress
Ext reg. DE next adrs
Rsv reg. except AF, DE

17 .COUNT
Function 文字長を計算（先頭から 00 H まで）
Ent reg. DE string pointer
Ext reg. A string length (end-code is 00 H)
Rsv reg. except AF

18 .ADDP0
Function add offset to POOL, VARST, STRST, VARED, and TMPEND
Ent reg. DE offset
Ext reg. non
Rsv reg. except HL, Flag

19 .ADDP1
Function add offset to VARST, STRST, VARED, and TMPEND
Ent reg. DE offset
Ext reg. non
Rsv reg. except HL, Flag

1A .ADDP2
Function add offset to STRST, VARED, and TMPEND
Ent reg. DE offset
Ext reg. non
Rsv reg. except HL, Flag

1B .ERRX
Function エラーメッセージの表示.
Ent reg. A error code. if bit7=1 then display device-name too.
Ext reg. non
Rsv reg. non

1C .?ADCN
Function MZ-ASCII よりディスプレイコードに変換.
Ent reg. A mz-ascii-code
Ext reg. A mz-display-code
Rsv reg. except AF

1D .?DACN
Function ディスプレイコードより MZ-ASCII へ変換.
Ent reg. A mz-display-code
Ext reg. A mz-ascii-code
Rsv reg. except AF

1E .?PNT1
Function x, y 座標より, VRAM アドレスの算出.
Ent reg. L x-coordinate

H y-coordinate
Ext reg. HL VRAM adrs
Rsv reg. except AF, HL

20 .BELL
Function ベル音を鳴らす．MUSIC 実行中に行なうと MUSIC が止まる．
Ent reg. non
Ext reg. non
Rsv reg. except AF

21 .PLAY
Function 音楽を演奏する..PLAY をコールする前に，.MCTRL を使用して現在の音が終わるまで待つ必要がある．また，.PLAY をコールした後に，.MCTRL を使用して演奏を開始する必要がある．つまり，次の様なシーケンスになる........MCTRL (3) .PLAY .PLAY .PLAY .MCTRL (1)
Ent reg. A channel (0～7), 3 and 7 is noise.
if A＝FFH then all-channel.
DE music data pointer (MML intermediate code)
Ext reg. non
Rsv reg. non

22 .SOUND
Function PSG を直接コントロールする．BASIC の SOUND 命令に対応．
Ent reg. if A＞＝80 H then PSG register set. BASIC command SOUND＝(r, d)A (register No.)＋80H
DE data
if A＜80H then sound out. BASIC command SOUND n, length
A 音階
DE 音長（10 ms 単位）
Ext reg. non
Rsv reg. non

23 .MCTRL
Function MUSIC に関して以下の動作を行なう．
Ent reg. B＝0 music init
B＝1 .PLAY start
B＝2 music stop
B＝3 music wait
Ext reg. non
Rsv reg. non

25 .TIMRD
Function time read
Ent reg. non
Ext reg. DE time (second)
A AM／PM……AM のとき
Rsv reg. except AF, DE

26 .TIMST
Function time set
Ent reg. DE time (second)
A AM／PM……AM のとき
Ext reg. non
Rsv reg. except AF

28 .CLRIO
Function イニシャライズ．
Ent reg. non
Ext reg. non
Rsv reg. non

29 .SEGAD
Function get segment adrs
Ent reg. A segment No.
Ext reg. HL segment adrs
if not exist then CF:＝1
Rsv reg. except HL, Flags

2A .OPSEG
Function open segment. No-check if already exist.
Ent reg. A segment No.
DE length
Ext reg. HL segment adrs
Rsv reg. except AF, HL

2B .DLSEG
Function delete segment. if no-exist then NOP.
Ent reg. A segment No.
Ext reg. non
Rsv reg. all

2C .DEVNM

Function interpret device-name. uses KEYBUF.

Ent reg. DE pointer
B length

Ext reg. HL pointer (next)
DE device table
A channel

Rsv reg. non

2D .DEVFN

Function interpret device-name and file-name

Ent reg. DE pointer
B length

Ext reg. non

Rsv reg. non

2E .LUCHK

Function check logical-unit-number

Ent reg. A lu

Ext reg. A 1:R, 2:W, 4:X. if not-opened then CF:=1

Rsv reg. except AF

2F .LOPEN

Function open file for LOAD, MERGE, CHAIN, RUN, VERIFY. call after .DEVFN

Ent reg. non

Ext reg. non

Rsv reg. non

30 .LOADF

Function Load file. must call after .DEVFN and .LOPEN

Ent reg. HL loading adrs

Ext reg. non

Rsv reg. non

31 .SAVEF

Function Save file. must call after .DEVFN

Ent reg. DE save adrs (length, load-adrs, exec-adrs is ELMD20, 22, 24)

Ext reg. non

Rsv reg. non

32 .VRFYF

Function Verify files. must call after DEVFN, .LOPEN

Ent reg. HL adrs

Ext reg. non

Rsv reg. non

33 .RWOPN

Function ROPEN or WOPEN file. must call after .DEVFN
(ZRWX)=1...ROPEN,
(ZRWX)=2...WOPEN

Ent reg. non

Ext reg. non

Rsv reg. Except AF

34 .INSTT

Function input start. called each INPUT statement.

Ent reg. non

Ext reg. non

Rsv reg. except AF

35 .INMSG

Function input message from file

Ent reg. DE input buffer

Ext reg. B length

Rsv reg. only DE, HL

36 .INDAT

Function read 1 item from buffer

Ent reg. HL data pointer
DE input buffer
C separater (normally 00H or ":")

Ext reg. HL next pointer
B length

Rsv reg. non

37 .PRSTR

Function print string into file

Ent reg. DE string adrs
B string length

Ext reg. non

Rsv reg. except AF

38 .CLKL
Function CLOSE/KILL file (s)
Ent reg. A lu, if A=0 then all-files.
B=0 KILL
B< >0 CLOSE
Ext reg. non
Rsv reg. except AF

39 .DIR
Function directory. must call after .DEVNM
calling sequene.... .DEVNM, DIR (0), DIR (lu)
Ent reg. if A=0 then read directory
else output directory (A=output lu)
Ext reg. non
Rsv reg. except AF

3A .SETDF
Function set default device
Ent reg. DE device table adrs
A unit No.
Ext reg. non
Rsv reg. all

3B .LSALL
Function LOAD ALL/SAVE ALL
Ent reg. A=0 LOAD ALL
A=1 SAVE ALL
Ext reg. non
Rsv reg. non

3C .FINIT
Function I/O driver init (BASIC INIT command). must call after .DEVNM
Ent reg. non
Ext reg. non
Rsv reg. non

3D .DELET
Function delete file. must call after .DEVFN
Ent reg. non
Ext reg. non
Rsv reg. except AF

3E .RENAM
Function rename file. must call after .DEVFN
Ent reg. DE new name pointer
B new name length
Ext reg. non
Rsv reg. except AF

3F .LOCK
Function LOCK/UNLOCK file. must call after .DEVFN
Ent reg. non
Ext reg. non
Rsv reg. except AF

40 .RECST
Function random-file record-number set
Ent reg. DE record number
Ext reg. non
Rsv reg. all

41 .INREC
Function input from random-file
Ent reg. DE input buffer
Ext reg. B length. normally 32. if eof then 0.
Rsv reg. only DE, HL.

42 .PRREC
Function output into random-file
Ent reg. DE input buffer
B length.
Ext reg. non
Rsv reg. except AF

43 .ERCVR
Function error recovery and FD/QD motor off
Ent reg. non
Ext reg. non
Rsv reg. non

44 .SWAP
Function SWAP file subroutine
swap out calling sequence
B=0; .DEVFN;
SWAPDV=DE; SWAPCH=A;
ELMD20=text length;
A=0; .SWAP;

load new file;
swap file recovery sequence
DE=SWAPDV; A=SWAPCH;.
SETDF;
A=FFH; .SWAP;
Ent reg. A swap out or recovery flag
Ext reg. non
Rsv reg. non

45 .CLS
Function clear screen
Ent reg. A CRT page number
Ext reg. non
Rsv reg. except AF

graphics package

46 .POSCK
Function 指定された x, y 座標がスクリーン内かどうかチェック，スクリーン外のときは ER 03
Ent reg. DE' x-coordinate
HL' y-coordinate
Ext reg. non
Rsv reg. except AF

47 .POSSV
Function save position data to work area (POINTX, POINTY).
Ent reg. DE' x-coordinate X 座標を Pointx に
HL' y-coordinate Y 座標を Pointy に SAVE
Ext reg. non
Rsv reg. all

48 .PSET
Function SET または RESET に対応.
Ent reg. DE x-coordinate
HL y-coordinate
A<>0 SET
A=0 RESET
Ext reg. non
Rsv reg. non
モニタコーリングシーケンス

49 .LINE
Function LINE または BLINE に対応.
Ent reg. DE' start x-coordinate
HL' start y-coordinate
DE end x-coordinate
HL end y-coordinate
A<>0 LINE
A=0 BLINE
Ext reg. DE' end x-coordinate
HL' end y-coordinate
Rsv reg. non

4A .PATTR
Function PATTERN に対応.
Ext reg. DE string adrs
A height
B string length
H direction (0:upward, 1:downward)
Ext reg. non
Rsv reg. non

4B .BOX
Function Box に対応.
Ent reg. DE x-coordinate 1
HL y-coordinate 1
DE' x-coordinate 2
HL' y-coordinate 2
CF=0 BOX only
CF=1 BOX fill
A fill color
Ext reg. non
Rsv reg. non

4C .PAINT
Function graphic PAINT
Ent reg. (coordinate & POINTX, POINTY)
HL color data adrs (境界色)
B color data count
Ext reg. non
Rsv reg. non

4D .CIRCL
Function CIRCLE に対応.
Ent reg. (center is POINTX, POINTY)
DE' start x-coordinate

HL' start y-coordinate
DE end x-coordinate
HL end y-coordinate
IX R
C' ratio flag
C'=0 ratio=0
C'=1 ratio<1, B'=256 * ratio
C'=2 ratio>1, B'=256／ratio
A angle flag
A =83 H start angle—end angle< -2π
A=82H -2π<start angle−end angle< $-\pi$
A=81H $-\pi$<start angle−end angle<0
A=00H start angle=end angle
A=01H 0<start angle−end angle< π
A=02H π<start angle−end angle< 2π
A=03H 2π<start angle−end angle

Ext reg. non
Rsv reg. non

4E .POINT
Function POINT関数に対応.
Ent reg. DE x-coordinate
HL y-coordinate
Ext reg. A palette code.
if out-of screen then A=FFH.
Rsv reg. non

4F .HCPY
Function ハードコピーを行なう.
Ent reg. A HCOPY parameter (1～3).
Ext reg. non
Rsv reg. non

モニタコーリングシーケンス

3-2 MZ-1500のハードウェア

3-2-1 MZ-1500の構成

ここではまず図3・1のMZ-1500のブロック図を使って解説を進めていきましょう.

(1) 各ブロックの概要

(a) QD（クイックディスク）部

CPUとQDのデータの受け渡しには,Z 80 SIO/0（LH 0084 A）を使用して行なっています.

QD I/FはSIOとQDドライブの間にあって,データの変調,復調を行なっています.

(b) LPT部

CPUとプリンタ(LPT)のデータの受渡し,プリンタの制御にはZ 80 PIO(LH 0081 A)を使用しています.

また,PIOはタイマIC（8253）からの割込み制御も行なっています.これは8253の割込み要求に対し,PIOが8253に代りベクトルを発生させベクトル割込みを実現させるものです.

LPT I/FはPIOとプリンタの間にあって信号のドライブ,MZプリンタとセントロプリンタの切換えを受け持っています.

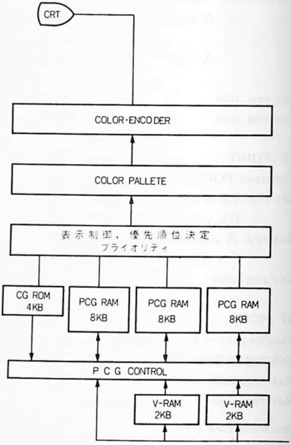
(c) TIMER (8253) 部

TIMERはサウンド発生,24時間時計として動作しています.音程,時間セットはCPUからデータを書込むことで行なわれます.このデバイスはメモリ上にマッピングされています(E 004 H～E 007 H).PIT(プログラマブル・インターバルタイマ).

(d) SOUND部

サウンドは2つのPSG（プログラマブル・サウンドジェネレータ）を使っています.デバイスはSN 76489 ANです.この2つのPSGを使って右チャネル,左チャネル独立したステレオ音楽を実現しています.本体内蔵スピーカには右チャネルと左チャネルを合成した音をアンプを通して出力しています.

PSGには外部入力端子があり,その端子には8253のサウンド出力,オプションのボイス出力が



接続されています．

(e) VOICE 部

ボイス部はオプションのボイスカード(MZ-1M08)を接続できるようインターフェイス部のみ内蔵しています．

(f) KEY, CMT 部

CPU と KEY, CPU と CMT のデータの受け渡しに 8255（PPI:プログラマブル・パラレルインターフェイス）を使用しています．

キーの制御には 8255 のポート PA_0～PA_3と PB_0～PB_7を使用しています．

カセット(CMT)の制御には 8255 のポート PC_1, PC_3～PC_5を使用しています．

KEY I／F はキーストローブのデコード, CMT I／F は，カセットデータの変調，復調，モータ制御を行なっています．

このデバイスはメモリ上にマッピングされています(E000～E003)．

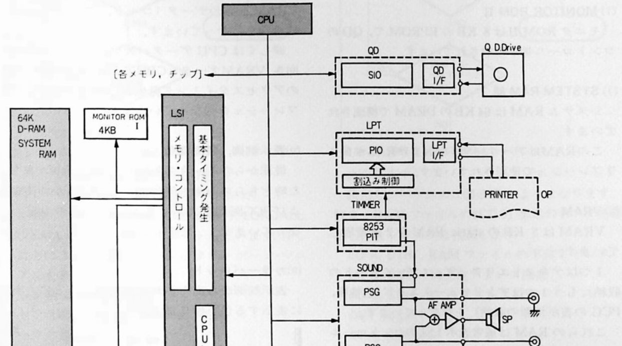


図3・1　MZ-1500 のブロック図

(g) JOY 部

JOY 部はオプションのジョイスティック(MZ-1X03)を接続するためのインターフェイスで，2 個持っています．

(h) MONITOR ROM I

モニタROM I は，MZ-700とほぼ同様の内容で 4 KB の EPROM です．

このROMをCPUがM1サイクルにおいてアクセスした時 1 wait 挿入されます．

(i) MONITOR ROM II

モニタ ROM IIは 8 KB の EPROM で，QD のコントロール等が収納されています．

(j) SYSTEM-RAM 部

システム RAM は 64 KB の DRAM で構成されています．

このRAMはアーリーライトおよびRASオンリリフレッシュで使用されています．

(k) VRAM

VRAM は 2 KB の static RAM を 2 個使用しています．

1 つはテキストエリア（ディスプレイコードの収納），もう 1 つはアトリビュートエリア（色情報，PCG の表示情報の収納）を行なっています．

これらの RAM は通常表示 LSI のコントロール下にあり，CPU がアクセスを行なった時のみ CPU にコントロールが移ります．

(l) CG ROM

キャラクタジェネレータROMは，文字パターンを収納している 4 KB のマスク ROM です．

この ROM も通常 PCG コントロールの管理化にあり，CPU がこの ROM に対して，アクセスを行なった時のみ CPU の管理に移ります．ただし表示中は CPU の管理下には入りません．これは表示画面が乱れるのを防ぐためです．現実には CPU に対し wait を発生しています．

(m) PCG RAM

Programmable Character Generatorの略でソフトウェア定義可能なキャラクタジェネレータで，8 KB の擬似スタティック RAM を使用しています．この RAM は Red, Green, Blue の 3 個あります．これにより 8 色を表現しています．

擬似スタティック RAM はリフレッシュを必要とする以外はスタティック RAM とほぼ同様に使用できます．リフレッシュは表示に合わせて行なっています．

この RAM も通常 PCG コントロールの管理下にあり CPU がアクセスした時のみ CPU の管理下に移ります．

(n) PCG コントローラ

PCG コントローラは PCG データバスの管理，VRAM からのデータ（コード，アトリビュート）の管理を行なっています．

詳しくは CPU データバスゲートのイネーブル，向き，VRAMデータとCPUアドレスの切換え，PCG のアクセスタイミング発生および PCGRAM のリフレッシュを行なっています．

(o)表示制御，優先順位

従来からのキャラクタ画面と PCG 画面が重なった時どちらの画面を優先的に表示するかの決定およびPCG画面を表示するかしないかの決定を行なっています．

(p)カラーパレット

表示制御から出力される色情報をもとに，実際に表示する色を作ります．

3-2-2
RAMファイル

RAM ファイルの使用例

RAM ファイルとはメモリをひとつの外部記憶装置として使用するもので，RAM ファイルを使うとメモリをフロッピーディスクやカセットテープのように使うことができます．(ただし電源を OFF にすると，記憶されていた内容は消去されます．)

プログラム中で使用するデータの書込みや読出しを高速で行うことができるので，じょうずに使うと効率のよいデータ処理などができます．RAM ファイルの仕様についてはつぎのようになっています．

RAM：64 KB（M 5 K 4164 P-20 相当品× 8）
機能：READ, WRITE アドレス，オートインクリメント
その他：本体 I／O スロット収納形

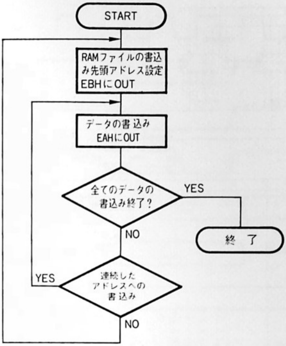


図3・2　RAMファイルの書込みシーケンス

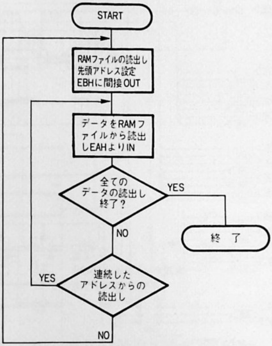


図3・3　RAMファイルの読出しシーケンス

I／Oアドレス EA, EB 固定
（EA…データ　EB…アドレス）

RAM ファイルは BASIC では INIT, LOADALL, SAVEALL などの命令をサポートしています. 詳しいことは MZ-1500 のベーシックランゲージ・マニュアルを参照して下さい.

マシン語による使用法

アドレス指定　I／Oアドレス EBH
データの書込みと読出し　I／Oアドレス EAH

書込みシーケンスは図3・2のようになります.

例1：RAMファイルの, $D3C5番地にデータ$41を書込みます.

```
LD A, C5H RAM ファイルの下位アドレス設定
LD B, D3H 同上位アドレス設定
LD C, EBH I／Oアドレス
OUT (C), A    必らず間接OUT命令を使う
LA A, 41H     } OUT(EBH), A は不可
OUT (EAH), A  } データの出力
```

アドレス設定は必らず間接 OUT を利用してください. これらをまとめると

```
3E C5
06 D3
0E EB
ED 49
3E 41
D3 EA
```

になります.

読出しシーケンスは図3・3のようになります.

例2：RAMファイルの$CD01番地のデータを読出し, Aレジスタに代入します

```
LD A, 01H   RAM ファイルの下位アドレス
LD B, CDH   同上位アドレス設定
LD C, EBH   I／Oアドレス設定
OUT (C), A
IN A, (EAH)
```

これらをまとめると

```
3E 01
06 CD
0E EB
ED 49
DB EA
```

になります.

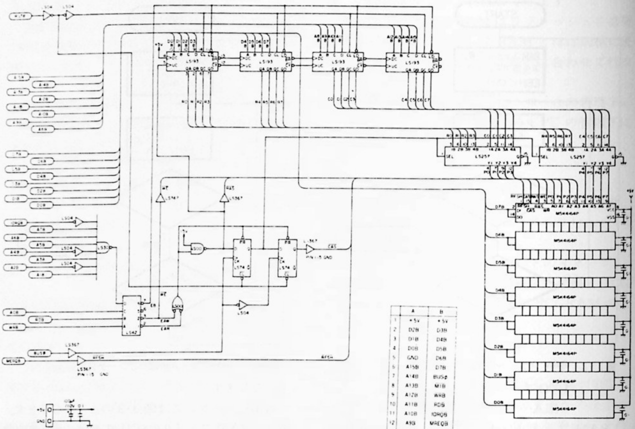
RAMファイルの構成

RAMファイルのブロック図を図3・4に示します. では各部について説明しましょう.

① I／Oアドレスデコーダ

I／Oアドレスデコーダ部は, このカードのI／Oアドレスを決定する回路で MZ-1R 18 では 0EAH, 0EBH($EA, $EB)に設定されています.

② DRAM部



	A	B
1	+5V	+5V
2	D2B	D3B
3	D1B	D4B
4	D0B	D5B
5	GND	D6B
6	A15B	D7B
7	A14B	BUSφ
8	A13B	M1B
9	A12B	WRB
10	A11B	RDB
11	A10B	IORQB
12	A9B	MREQB
13	A8B	GND
14	A7B	
15	A6B	
16	A5B	
17	A4B	RSTB
18	A3B	
19	A2B	
20	A1B	
21	A0B	
22	GND	GND
	部品面	半田面

図3・4　RAMファイルの構成

64 K×1のRAMを8個利用して64 KBの記憶容量を持っています．

RAMは，リフレッシュ端子を持ったものでRAMを交換する時には注意が必要です．

③　タイミングコントロール部

DRAMの読出し，書込み時に必要な信号を生成します．

$\overline{\text{RAS}}$……DRAMのロウ(行)アドレス・ストローブ信号

$\overline{\text{CAS}}$……DRAMのカラム(列)アドレス・ストローブ信号

SEL……ロウアドレスとカラムアドレスのマルチプレクス（切換）信号

$\overline{\text{WE}}$……DRAMのライト・イネーブル信号

$\overline{\text{RFSH}}$……DRAMのリフレッシュ信号，リフレッシュはCPUの$\overline{\text{MERQ}}$に同期して行なわれます

$\overline{\text{LOAD}}$……DRAMのアドレスをアドレスカウンタにロード（セット）します

COUNT……DRAMからのデータ読出しまたは書込みごとに発生する信号で，この信号によりアドレス・カウンタをインクリメント（+1）します．

④　16 bit アドレスカウンタ

プリセット可能な16 bit長のカウンタでDRAMのアドレスを発生します．

このカウンタは，オートインクリメント機能に利用しています．

⑤　2 TO 1 データセレクタ

16 bitのアドレスカウンタの出力をマルチプレクスしてDRAMのロウ(row)アドレス，カラム(column)アドレスを作ります．

タイミング解析

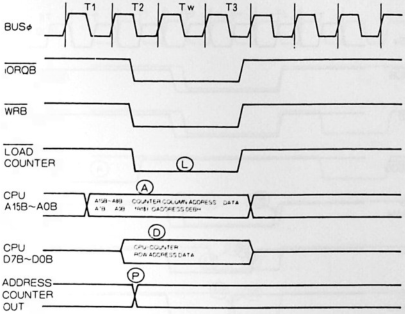


図3・5 A
DRAMへのアドレス設定
タイミングチャート

RAMファイルのアドレス設定のタイミングチャートは図3・5Aのようになっています．以下，それを説明します．

①CPUのアドレス・バスのA_7～A_0にアドレスカウンタのI／Oアドレス$EA，(0EAH)が出力されます．Ⓐ

②つづいて，CPUからの$\overline{IORQB}$，$\overline{WRB}$が"L"レベルになります．これにより，$\overline{LOAD}$が"L"になります．Ⓛ

③この時A15B～A8Bにはカウンタにセットするデータの上位8 bitが出力され，D7B～D0Bには，カウンタにセットするデータの下位8 bitがCPUより出力されています．Ⓐ，Ⓓ

④$\overline{LOAD}$が"L"になるとCPUより出力されているA15B～A8BとD7B～D0Bのデータがカウンタにセットされます．Ⓟ

DRAMへのデータ書込み

図3・5BにDRAMへのデータ書込みのタイミングチャートを示します．

① DRAMへのデータ入力ポート$EB(0EBH)に対してデータを出力すると，$\overline{IORQB}$と$\overline{WRB}$が"L"レベルになります．

② これによりDRAMの$\overline{RAS}$が立下がり，立下がりエッジで，アドレスカウンタの下位8 bitをDRAM ROWアドレスとして取込みます．(E1)

③ 次に$\overline{RAS}$をTWの立上がりでサンプルしてSELを作ります．(E4)

SEL信号は，2 TO 1データセレクタのセレクト端子に接続されています．SELが"L"の間DRAMにカラムアドレス(アドレスカウンタの上位8 bit）が与えられています．

④ $\overline{CAS}$はSEL信号をTWの立下がりエッジでサンプルして作られています．(E3)

DRAMは$\overline{CAS}$の立下がりでアドレスカウンタの上位8 bitを，DRAMのカラムアドレスとして取込みます．

この時，DRAM $\overline{WE}$が"L"レベルとなっているので(L1)アーリーライト(注)になりD7B～D0Bのデータを RAMに書込みます．

⑤ $\overline{IORQ}$またはWRが"L"レベルから"H"レベルに変化すると，$\overline{RAS}$，$\overline{CAS}$，SELはHになり，DRAMのアクセスを終わります．

$\overline{RAS}$立上がりエッジでアドレスカウンタをカウント・アップします．(E2)

⑥ $\overline{RFSH}$は$\overline{MREQ}$に同期して"L"になりDRAMのリフレッシュを行ないます．(L2)

注） アーリーライト

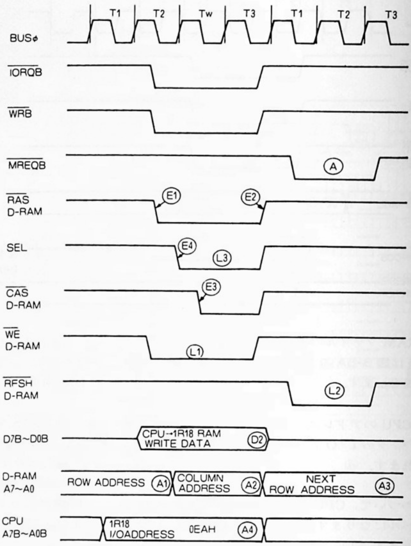


図3・5 B
RAMファイルのデータ書込み
タイミングチャート

DRAMのライトサイクルには，$\overline{CAS}$を"L"にした後，$\overline{WE}$を"L"にするレイトライトサイクルと$\overline{WE}$を"L"にし次にCASを"L"にする，アーリーライトサイクルがあります．

この2つの大きな違いは，レイトライトサイクルでは，選択されたアドレスのデータが出力端子に出力されることと，アーリーライトサイクルでは，出力端子は，Hi-Z（ハイインピーダンス）になっていることです．

出力端子がHi-Zになっていると出力端子と入力端子を接続して，バスの共通化ができます（図3・6）．

DRAMからの読出し

図3・7はデータの読出しのタイミングチャートです．

① DRAMのデータ入出力ポート$EB(0EBH)に対し入力命令を実行すると$\overline{IORQB}$と$\overline{RDB}$が"L"レベルになります．

② これによりDRAMの$\overline{RAS}$が立下がります．この立下がりエッジでアドレスカウンタの下8 bitをDRAMのROWアドレスとして取込みます．(E1)

③ SEL信号はアドレスカウンタの16 bitの出力を8 bitずつにマルチプレクスするためのものです．(L3) (A2)

この信号は$\overline{RAS}$信号をTWの立上がりエッジでサンプルすることにより得られます．(E4)

④ $\overline{CAS}$信号はSEL信号をTWの立下がりエッジでサンプルすることで得られます．

DRAMは，$\overline{CAS}$の立下がりエッジでマルチプレクスされて出て来たアドレスカウンタの上位8 bitをカラムアドレスとして取込みます．(E3)

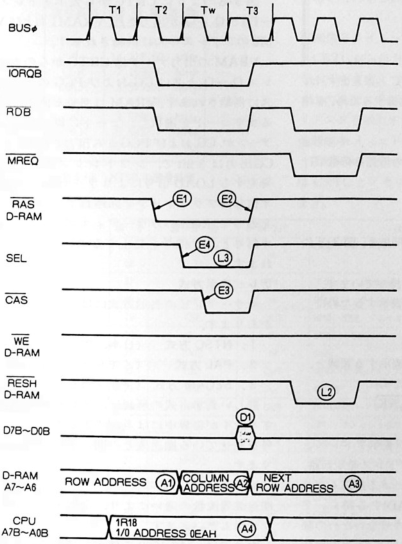


図3・7
データの読出し
タイミングチャート

⑤ DRAMの出力は,$\overline{\text{CAS}}$の立下がりから一定時間(アクセスタイム)後,有効データがD7B~D0Bに出力されます.(D1)

⑥ CPUはデータバス上のデータをT3の立下がりエッジで取込みます.(D1)

⑦ $\overline{\text{IORQB}}$または$\overline{\text{RDB}}$が"L"レベルから"H"レベルに変化すると$\overline{\text{RAS}}$,$\overline{\text{CAS}}$, SELは"H"レベルになり,ダイナミックRAMへのアクセスを終わります.

⑧ $\overline{\text{RAS}}$の立上がりエッジでアドレスカウンタをカウントアップします.(E2)

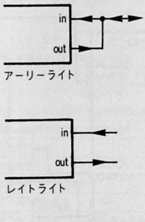


図3・6 バスの共通化

3-2-3

CRTC

CRT コントローラ(以後 CRTC と書きます)は, MZ-1500 のテレビ画面表示を制御するため, 専用 IC(M60719)を使用しています.

また, この IC は後述するメモリコントローラとしても使用しており, MZ-1500 の性能を特徴づける重要な IC です.

CRTC と VRAM, CG の関係

CRTC と VRAM と CG との関係を, 図 3・8 に示します.

CRTC は, おもに次の機能を持っています.

① テレビ画面に画像データを表示するために
a 水平同期信号($\overline{\text{HSYNC}}$)
b 垂直同期信号($\overline{\text{VSYNC}}$)

② テレビ画面の画像データを表示する領域と, そうでない領域を分離するために
a 水平ブランキング信号($\overline{\text{HBLNK}}$)
b 垂直ブランキング信号

③ 画像データを表示する位置(表示アドレスといいます)を発生する表示アドレス発生回路.

④ VRAM に, CPU からのデータを書込んだり(WRITE)読出したり(READ)する時に, テレビ画面にチラツキを発生させないための制御回路と CPU アドレスと表示アドレスを混合する回路.

図 3・8 のように CRTC から表示アドレス, P_0 ~P_{10}と Q_1~Q_3が出力され, VRAM1 および VRAM2 のアドレスバスに供給されます.

VRAM の出力データは CRTC からの表示アドレス Q_1~Q_3と共に CG および PCG のアドレスバスに供給されます. VRAM はテレビ画面に表示する文字コードを選択し, コードに従ったドットパターンが, CG および PCG から出力されます. PCG, CG出力は 8 bit で, シフトレジスタに CRTC の発生する LOAD 信号によりラッチされます. ビデオ信号はプライオリティ回路およびパレット回路を通って, RGB のカラーディスプレイおよびビデオ信号と, RF 信号を発生するエンコーダに出力されます.

テレビ表示方式

カラーテレビの表示方式には, 次の 3 つの方式があります.

1. NTSC 方式 (日本, アメリカなど)
2. PAL 方式 (イギリス, ドイツなど)
3. SECAM 方式 (フランスなど)

詳しい表示方式の解説は, テレビの専門書にゆずりますが世界中には上記の表示方式とビデオ信号を乗せている搬送波との間に無数の組合せがあります.

同じ NTSC 方式の日本とアメリカ(USA)でも, 搬送波周波数の違いにより, 全体としては異ったシステムといえます.

表 3・3 は, MZ-1500 で使用している信号の周波数です. 各信号名称は図 3・8 の CRTC の信号端子

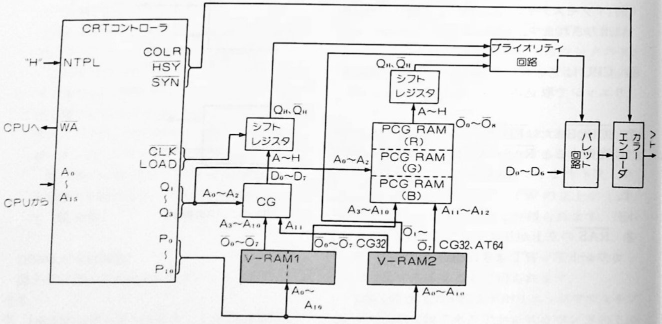


図3・8 表示部ブロック図

信号名称	信号周波数
N/P	"H"
φ	14.318180 MHz
$\overline{\text{CLK}}$	7.159090 MHz
COLR	3.579545 MHz
LOAD	0.894886 MHz
$\overline{ø}$	3.579545 MHz
$\overline{\text{HBLK}}$	15.6997 kHz
$\overline{\text{VBLK}}$	59.9227 Hz

表3・3　NTSC の信号仕様

名称と合わせています.

同期信号タイミング

CRTCから出力する同期信号は図3・9のようになっています.

この表示タイミングは日本でのNTSC方式のタイミングを満たすように設計しています.

信号端子

CRTCの信号を表3・4に示します. カスタムICはCRTCとメモリコントローラの機能を備えています.

端子	信号名	IN OUT	機能
1 〜 16	A 0 〜 A 15	IN	CPUアドレスバス
17	Φ	IN	クロック (14.3MHz)
18	ø	IN	CPUクロック (3.58MHz)
19	$\overline{\text{CSE}}$	OUT	8255, 8253, ジョイステックチップ・イネーブル
20	CL	IN	GND
21	$\overline{\text{GATE}}$	IN	GND
22	$\overline{\text{CSO}}$	OUT	MONITOR ROMチップィネーブル
23	5 V	—	電　源
24	$\overline{\text{RAS}}$	OUT	D—RAMローアドレスセレクト←メインRAMの選択
25	$\overline{\text{RFSH}}$	IN	CPUリフレッシュ信号
26	ø 0	OUT	CPUクロック生成信号 (3.58MHz) ←14.3MHzを分周して出力
27	$\overline{\text{MREQ}}$	IN	CPUメモリリクエスト
28	$\overline{\text{IORQ}}$	IN	CPU　I/Oリクエスト
29	$\overline{\text{RD}}$	IN	CPUリード
30	$\overline{\text{WR}}$	IN	CPUライト
31	$\overline{\text{RESET}}$	IN	リセット
32	SEL	OUT	DRAMのローアドレス、カラムアドレス切換信号
33	$\overline{\text{VBLK}}$	OUT	垂直ブランキング信号 (CRT)
34	$\overline{\text{HBLK}}$	OUT	水平ブランキング (CRT)
35	$\overline{\text{WA}}$	OUT	CPUウェイト信号
36	COLR	OUT	カラー副搬送波 (3.58MHz : NTSC)
37	$\overline{\text{PRC}}$	OUT	プリンタI/Oアドレスセレクト信号　FEH or FFH
38 〜 40	Q 1 〜 Q 3	OUT	表示 : アドレスデータ出力 (P_0〜P_{10}と共に表示アドレスをCG—ROMへ指定する)
41	N／P	IN	NTSC、PAL両方式切換 (NTSC=H)
42	BLNK	OUT	時間発生用クロック
43	$\overline{\text{HSY}}$	OUT	水平同期信号
44	ABC	OUT	8文字表示アドレス信号
45	LOAD	OUT	文字・表示開始信号
46 〜 52	P 0 〜 P 6	OUT	表示アドレス信号　V—RAMへ送られる
53	GND	—	接　地
54 〜 57	P 7 〜 P 10	OUT	表示アドレス信号
58	S 157	OUT	V—RAMの表示アドレス、CPUアドレスの切換信号
59	$\overline{\text{SYN}}$	OUT	垂直同期信号
60	$\overline{\text{CLK}}$	OUT	文字表示用シフトレジスタクロック

表3・4
CRTC の信号

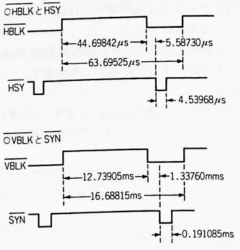


図3・9　CRTC からの同期信号

ブロック図

カスタム IC のブロック図を図 3・10 に示します。CRTC部は，ほとんどがカウンタで構成されています。

プライオリティ回路

MZ-1500 は，ROMCG と，PCG の重ね合せが可能であるため，表示においてどちらを優先するかを決定する回路を必要とし，これをプライオリティ回路といいます。

この回路は，図 3・11A のようになっています。

プライオリティ設定回路

この回路は，PCG とキャラクタが重なった場合どちらを優先的に表示するかを決める回路です。$\overline{\mathrm{GMODE}}$ と $\overline{\mathrm{WR}}$ から成る LS74 のクロック入力 (C_K) は，$F0 ポートにデータを書込んだ時，立下

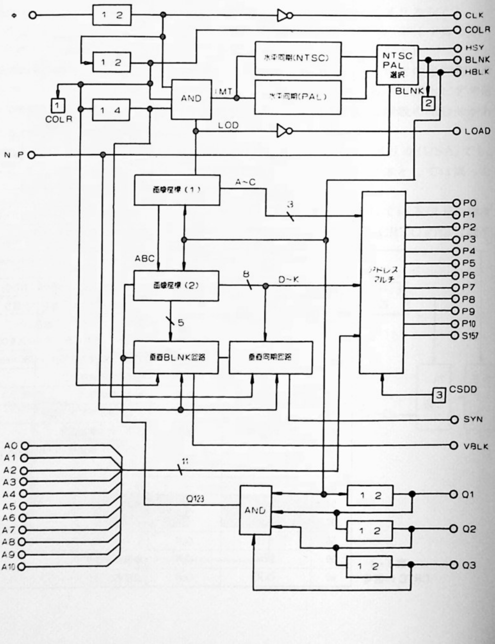


図3・10
CRTC のブロック図

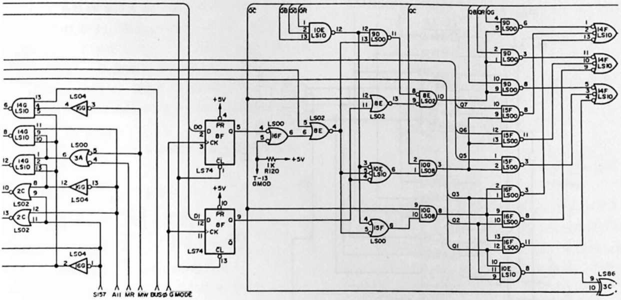


図3・11 A　プライオリティ回路部

がります．このクロックの立上がりで，データを取込み出力します．このデータは，PCGの表示の有無と，PCGとキャラクタの優先順位を決めます．

この書込みデータは，表3・5の通りです．

① $D_0=0$ のとき

LS74からの出力は“L”となり，キャラクタデータの通るIC10Gのゲートは開きキャラクタは表示されます．しかし，PCGデータの通るNANDゲート群(IC9D)は閉じられPCGは表示されません．

② $D_0=1$ のとき

LS74からの出力は“H”となり，VRAM2の後半($DC00～$DFFF)に書込まれたデータの，D_3 が“H”ならば，IC8Eの4ピンからの出力は“H”となりPCG表示モードとなります(DIP SW№3はOFF状態)．そして，PCGデータがあればIC10Eからの出力は“H”となります．すなわち，PCGとキャラクタが重なりPCGの方が優先度が高ければPCGは表示されます．また，PCGの方が優先度が低くてもフロントデータがなければ，PCGは表示されます．

③ $D_1=0$ のとき

LS74からの出力は“L”となり，IC10Eからの出力は“H”となる．すなわちフロントは表示されます．しかし，フロントデータがなくて，PCG表示モードであればPCGデータの通るNANDゲート群は開きPCG表示が可能となります．バックは，フロントデータ，PCGデータ共にない時に表示可能です．

④ $D_1=1$ のとき

LS74からの出力は“H”となり，IC8Eの13ピンからの出力は“L”となります．すなわち，PCG表

優先度	表示	D0	D1
キャラクタ		1	0
P C G		1	1

図3・11 B　画面の出力例

表3・5
プライオリティ設定の
書込みデータ

D7	未使用
D6	
D5	
D4	
D3	
D2	
D1	優先順位
D0	PCGの表示

D1	D0	優先度
0	0	PCGは表示しない
0	1	PCGを表示　フロント>PCG
1	0	PCGは表示しない
1	1	PCGを表示　PCG>フロント

表3・6　D_1，D_0 とプライオリティ

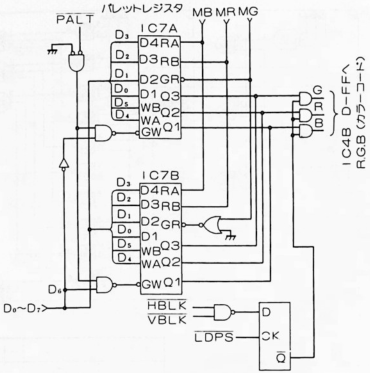


図3・12
パレット回路

示モードでなければ2つのIC10Gのゲートは開きキャラクタは表示可能となります.

以上の①〜④をまとめると表3・6のようになります.

画面の表示例(図3・11B)を参照してください.

パレット回路

通常テレビ画面に表示している色を変えるためには, VRAM上の書込みデータを書換えるわけですが, 色を変更するのに時間がかかり, 瞬時にして色換えができないことが長い間課題になっていました. パレット回路はこれを解決した回路で, あたかも, パレットの上で絵の具をまぜるように, テレビ画面の色を変えることができます.

以下, パレット回路について説明します.

図3・12を見て下さい.

この回路は, プライオリティ回路から出力される色情報(表示データ：MR, MB, MG)をもとに, 実際に表示する色を作る回路です.

パレットコードは初期状態ではカラーコード(実際に表示する色)と同じですが, パレットコードを変えることによって, メモリ内にある色データを書換えることなく瞬時に別の色に変えることができます.

この回路の中心は, IC7A, 7Bの, 2つのパレットレジスタです. パレットレジスタにデータを書

D_7	未 使 用
D_6	元 の 色
D_5	
D_4	
D_3	未 使 用
D_2	パレットコード
D_1	
D_0	

表3・7
パレット回路への書込みデータ

込む場合, $\overline{PALT}$が"L"でなければいけません. すなわち, $F1ポートにデータを書込めばよいことになります. その書込んだデータのうちD_6が"0"ならばIC7Aのパレットレジスタに, D_6が"1"ならばIC7Bのパレットレジスタに書込まれます.

その書込みデータは表3・7のとおりです.

プライオリティ回路から出力されるMG, MR, MBを3桁の2進数(MSB：MG, LSB：MB)と, またパレットコード(D_0〜D_2)を2進数と考えるとそれぞれ0〜7の数字となります.

初期状態では, MG, MR, MBによって決まるコードとパレットコードとは同じです. これらのデータのうちD_3については未使用ですが, "0"としておくのが望ましい処置です.

次の設定したパレットコードを読出すには，MGが"L"であればIC7Aが，MGが"H"ならばIC7Bが許可されます．書込んでおいたD_4，D_5とMB，MRによって決まるコードが一致すると，書込んでおいたD_0～D_2が出力されます．

これらの出力は，IC6AのANDゲートを通り，クロック入力($\overline{SCLK}$)の立上がりでIC4Bに取り込まれ出力されます．

IC6AのANDゲートが開くのは，表示期間中です．

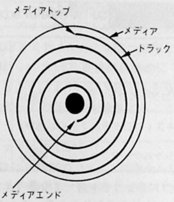


図3・13
QDメディアの構造

3-2-4 QDとQDコントローラ

QDの機能と複数ファイル管理方式

QDの外部記憶装置としての位置付けはデータレコーダとMFD(フロッピーディスク)の中間に当ります．スピードを除けば基本的な機能はデータレコーダと同一です．従ってQDもランダムアクセスファイルを扱うことはできません．しかしデータ転送速度101K bpsは MFDの実質データ転送速度よりも速く，システムソフトのローディングに関してはQDの方がMFDより適しているといえます．また，一般のデータレコーダでは，実使用面で採用できなかった(技術的には可能)メディア内のファイル登録内容の確認，および，登録されているファイルから任意のファイルを呼び出す等の処理を8秒以内で実行することができるのです．それにデータレコーダの場合，テープ内にすでにいくつかのファイルが存在していて，その後ろに新しいファイルを追加して登録する場合，いちいち頭出し作業を事前に行うか，テープカウンタでその場所を探しておかなければなりませんが，QDではそれらの作業をいっさい必要とせず自動的に処理することができます．

感触的にいえばQDはMFDに近いものを有していますので，MFDのローコスト版ではないかという見方もありますが，決してMFDと競合していくデバイスではなく，データレコーダの高速版と考えてもらうのが一番妥当であると考えます．

表3・8にQDの概略仕様を示します．この中にトラック数が1本とあります．MFDでは同心円状に複数個のトラックが形成されていますが，QDは

項目	仕様	備考
ディスク	2.8インチ	メカは片面であるが メディアは両面使用
記録方式	MFM	
記憶容量〔バイト〕	128K	64K×2
記録密度〔BPI〕	4410	
データ転送速度〔BPS〕	101K	
トラック密度〔TPI〕	59	
ヘッド数	1	
トラック数	1	スパイラル
ディスク回転数〔RPM〕	423	
消費電力	DC5V 0.3A	
リード／ライト時間	64K／8秒	ヘッド移動時間含む
周囲温度(℃)	10～40	
相対湿度(%)	30～80	湿中最高温度 29℃

表3・8
QDの概略仕様

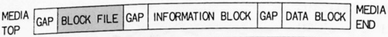


図3・14
ブロックファイルの位置

	INFORMATION BLOCK	DATA BLOCK
機械語プログラムファイル (OBJ)	1	1
BASICテキストプログラムファイル (BTX)	1	1
シーケンシャルデータファイル (BSD)	1	n

表3・9　ファイルの種類とブロック構成

図3・13のようにちょうどレコードの溝のように外周から内周に向けてうず巻状のトラックが1本だけ形成されているのです．データのSAVE／LOADはかならずディスクの最外周より始まります．QDの非動作時，ヘッドはメディアの最内周位置で停止しています．

パソコン側からQDに対して動作命令を出すと，メディアは回転を開始し，ヘッドはカムの作用により，いったん内周より外周へ素早く移動します．そして，うず巻状のトラックに沿ってヘッドは外周から内周へ移動してSAVE／LOADの動作を行ないます．この時のヘッドの動きもカムによって行なわれています．図3・13に示すようにメディアの最外周をメディアトップ，最内周をメディアエンドとし，うず巻状のトラックを一直線に引き伸ばして説明していきます．

メディア片面の記憶容量は64Kバイトです．実際には短いプログラムや，長いプログラムが混在して複数本がメディア片面に効率よく登録されなければ実用的とはいえず，MFDのようにセクタ管理方式を採用することは得策ではありません．セクタ単位にあらかじめフォーマットすれば，アンフォーマット時の64％に記憶容量がダウンしてしまうからです．QDは特殊なフォーマットを採用し，最大使用効率92％を得ています．ファイルのフォーマット自体はデータレコーダの場合とほとんど同じですが，QDではメディアに複数個のファイルを登録し，その管理を行なうために図3・14のようにメディアトップに現在登録されているファイル本数を書き込んでおく「ブロックファイル」というエリアを設けています．このブロックファイルの内容に従って，各ファイルのSAVE／LOADの操作が行われるのです．

1つのファイルはインフォメーションブロックとデータブロックの2種類にブロック化されています．各ブロックの間には，かならずギャップがあり，このギャップ部分でブロックの先頭を捜し

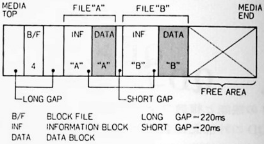


図3・15　ファイルの記録例

出していきます．ブロックファイルの内容は実際にはブロックの個数が書込まれるので，図3・14の場合その内容は2となります．

(1)ファイルの種類

QDで扱えるファイルの種類とブロック構成は表3・9の通りです．

OBJとBTXファイルは表3・9のように2ブロックで構成され，データブロックの大きさは可変です（最大64Kbyte）．

BSDファイルの場合，インフォメーションブロックは1つですが，データブロックはその大きさが1Kbyte単位で区切られており，複数個(n)となります．たとえばデータとして4.5Kbyteの大きさをもつBSDファイルのブロック数は，インフォメーションブロックを含めて6つということになります．

(2)メディアのイニシャライズ

新しいメディアを使用する前にメディアのイニシャライズを行わなければなりません．

イニシャライズを実行すると

①ブロックファイルの内容を，0として書込みをする

②メディアに欠陥がないかチェックを行う

という2つの動作を行います．従ってQDは2回（約16秒間）動作します．

(3)ファイルのSAVE

図3・15はメディアに2つのファイル“A”と“B”が登録されている状態を示しています．ブロック

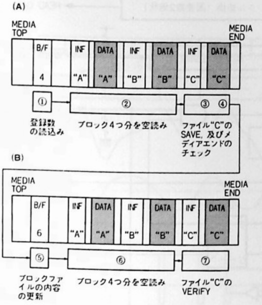


図3·16　ファイルの追加例

ファイルの内容はこの場合4となります．

このメディアに対して3つ目のファイル"C"をSAVEする場合，QDは次の2回の動作を行います．

1回目の動作(図3·16A)

①ブロックファイルの内容4を読込みます

②ブロックファイルの内容に従って4つ分のブロック，つまりファイル"A"，"B"を空読みします．この際ファイル名のチェックも行います（同じメディア内に同一のファイル名を付けることはできません）

③ファイル"B"の直後にファイル"C"のインフォメーションブロックとデータブロックを書込みます(ギャップも含む)

④メディアエンドまでにファイル "C" の書込みが終了したかをチェックします．ファイル"C"の容量が大きく，書込み途中でメディアエンドに達した場合は，QDのモータが停止すると共にエラーメッセージを表示(NO FILE SPACE)し，SAVEの動作を終了します．実際には，ファイル"C"の途中まで書込まれているわけですが，ブロックファイルの内容は更新されていないので，ファイル管理上このメディアはファイル"A"，"B"の2つだけが登録されている状態となります(図3·15と同じ状態)．

2回目の動作(図3·16B)

⑤ブロックファイルの内容を4から6へ更新して再登録します

⑥1回目と同様4つ分のブロックを空読みします

⑦ファイル"C"のベリファイ(VERIFY)を行います

以上でSAVE動作の終了です．SAVE時には書込み，ブロックファイルの内容更新，ファイルのVERIFYを行うために2回の動作となっています．このようにSAVE直後に自動的にVERIFYを実行することにより，SAVEミスをなくすことができ，データの信頼性を確保することができるのです．

(4)ファイルのLOAD

QDは複数個のファイルの中から任意の1ファイルのみを指定し，読出すことができます．この場合もブロックファイルの内容が重要な役割をしています．たとえば図3·16Bに示すメディアからファイル"B"をLOADしようとした場合，6つのブロック内のインフォメーションブロックよりファイル名"B"を捜し出し，そのインフォメーションブロックの内容に従って，データブロック"B"の内容を，パソコン側のメモリへ読出しますが，ファイル名に"D"を指定しLOADしようとした場合，6つのブロック内でファイル名"D"はありませんから，そのようなファイルは登録されていないとわかるわけです．ファイルをLOADするのはQDの1回の動作で終了しますので，約8秒ですが，LOADの順番は"A"，"B"，"C"ですのでファイル"A"の方が"B"，"C"よりもLOADに要する時間は速くなります．ファイル"A"が約10KバイトであればそのLOAD時間は約3秒程度となります．

(5)リトライ動作

SAVE／LOAD時の誤りに対してリトライを4回許しています，したがってLOAD時には最大5回，SAVE時は最大11回のQDの動作を伴う場合があります．

2．QD回路説明

(1) QD内部の回路説明とタイムチャート

図3·17にQDの回路図を簡略化して示します．リードアンプ部は磁気ヘッドが検出した入力電圧を増幅，微分，そしてゼロクロスコンパレータ回路を通すことによりメディア上の磁化反転位置に対応したロジックレベルの信号波形に変換する役目を持っています．続く周波数2倍回路(frequency dubler)で図3·18に示すREAD DATA信号に変換します．

WRITE DATAは周波数1/2回路(frequency divider)を介しヘッドの書込み電流の方向をコントロールします．ヘッドの書込み電流の方向を変えることによりメディア上の磁化の方向が変わり，

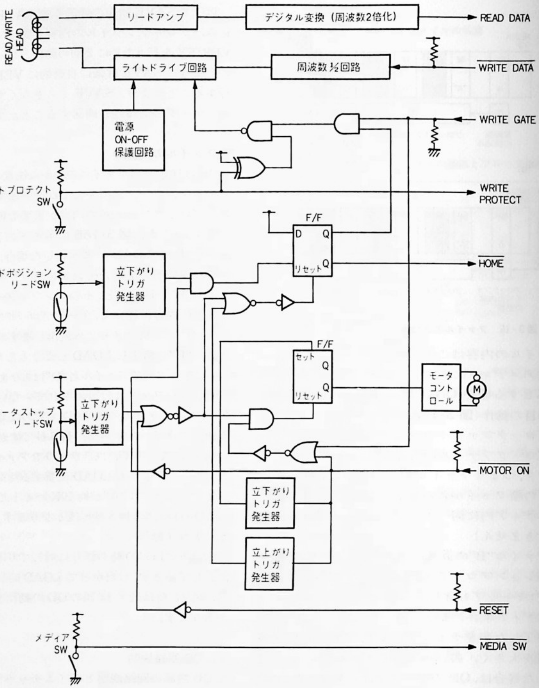


図3・17　QDの回路概略

WRITE DATAに対応した磁化反転をメディア上に分布させることができるのです.

QDをメディアにセットした状態では，メディアとヘッドは接触しています．通常のヘッド停止位置はメディア最内周のREAD／WRITE領域外となりますが，何らかの事情でヘッドが中途で停止している場合，電源をON／OFFしてヘッドに電流が流れるとメディアの記録内容が破壊される危険がありますので，電源ON／OFF時にはヘッドに電流を流さないようにする保護回路がついています.

書込み禁止は，QD側ではライトプロテクトSWによる検出，ソフト側ではWRITE GATE信号により行われます.

ヘッドポジションリードSWはメディアの最外周位置を検出するためのもので，この信号の立下りによりREAD／WRITEの動作がスタートします.

モータストップリードSWはモータ停止位置を検出するためのものです．書込み中にヘッドがモータ停止位置にきた場合，$\overline{\text{MOTOR ON}}$信号がLOWレベルであってもモータは停止します．モーターコントロール部では回転数安定化をはかるた

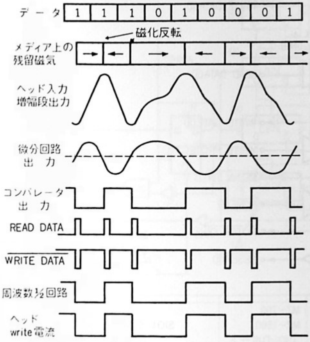


図3・18 リード／ライト時の各波形

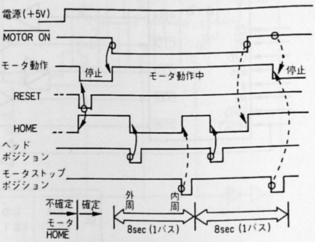


図3・19 QD回路のタイミングチャート

めにサーボIC回路がついています．

図3・19はQD回路のタイミングチャートを示しています．

$\overline{\text{RESET}}$　この信号がLOWになるとモータが強制ストップされ，$\overline{\text{HOME}}$信号がHighとなる．

$\overline{\text{HEAD POSITION}}$　信号の立ち下がりにより$\overline{\text{HOME}}$がLOWとなる．この信号を基準としてREAD／WRITE動作がスタートする．

MOTOR STOP POSITION　信号の立ち下がりにより$\overline{\text{HOME}}$がHighとなる．またこのとき$\overline{\text{MTON}}$がすでにHighであれば，モータを停止させる．$\overline{\text{MTON}}$がLOWでもWRGAがHigh(書込み中)であれば同じくモータを停止させる．

HOME　Highとなる条件…$\overline{\text{RESET}}$がLOW

MOTOR STOP POSITIONの立下がりまたは$\overline{\text{MTON}}$の立上がり

LOWとなる条件…HEAD POSITIONの立下がり

(2) QDインターフェイス回路説明

① FM記録方式とMFM記録方式

磁気記録方式としては，FM(Frequency Modulation)，MFM(Modified Frequency Modulation)のほかに，M^2FM，GCRなどがありますが，QDではFDDと同様にMFM方式を採用しています．QDの開発当初はFM方式で検討を行っており，記憶容量も32Kバイトと現在の半分だったのですが，

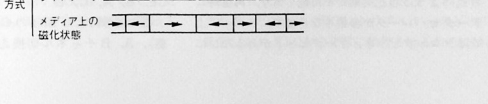


図3・20
FM方式とMFM方式

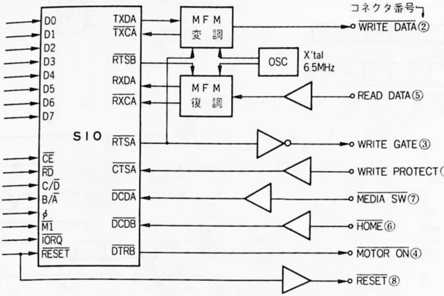


図3・21
QDインターフェイス
ブロック図

$\overline{CE}$	B/$\overline{A}$ (S 0)	C/$\overline{D}$ (S 1)	MZ-700 MZ-1500 でのI/Oポート	SIOレジスタ
0	0	0	$F 4	SIO　チャネルA　DATA
0	1	0	$F 5	SIO　チャネルB　DATA
0	0	1	$F 6	SIO　チャネルA　CWR
0	1	1	$F 7	SIO　チャネルB　CWR
1	−	−		SIOノットセレクト

表3・10
$\overline{CE}$, B/$\overline{A}$, C/$\overline{D}$のコントロール

64Kバイト化のためにMFM方式を採用したのです．この項ではFM, MFMの各記録方式について説明します．

FM方式ではデータの"1", "0"にかかわらず，かならずクロックビットがビットとビットの境界(ビットセルの先頭)に存在し，ビットセルの中央にデータビットがある場合を"1"，無い場合を"0"として記録・再生していきます．自分自身の信号，すなわちクロックビットで同期がとれるため，回転変動に対しての追従性が良いという利点をもっています．

一方MFM方式では，クロックビットが存在せずデータ"1"の場合はビットセルの中央にデータビットをおき，"1"の直後の"0"のビットセルにはデータビットをおかず，連続した"0"の場合，2つ目以降のビットセルの先頭にデータビットをおいていくという規則に基づいて記録再生していきます．MFM方式ではクロックビットが無いために，FM方式のように自己同期が不可能であり，再生時にデータセパレータが必要となります．メディア上ではクロックビット，データビットがあるたびに磁化反転されていきます．MFM方式の場合ビットセルには1つのデータビットがあるだけなので，FM方式に比べて磁化反転密度は半分で良く，逆にいえば同じ磁化反転密度においてはMFM方式ではFM方式の2倍の記録密度が得られるというわけです．図3・20はその具体的な例を示しています．

② QDインターフェイス回路

図3・21にQDインターフェイス回路のブロック図を示します．CPUとQDとはZ80SIOによってシリアルパラレル変換され，接続されています．

SIOはプログラムによりパラレルデータのシリアル変換およびシリアルデータのパラレル変換を行うと共に，各コントロール信号の入出力を行っています．SIOの各信号は次の意味を持っています(SIOはチャネルA, Bの2チャネルをもっています．)

$\overline{CE}$, B／$\overline{A}$, C／$\overline{D}$

この信号によりSIOのイネーブル(送受信可能状態)，A，Bチャネル切換え，データコントロール

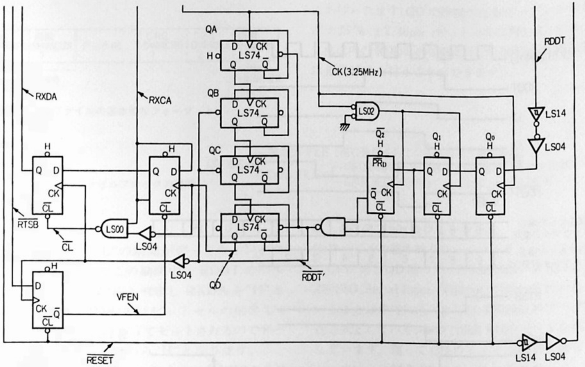


図3・22　MFM復調回路

切換えを行います(表3・10)

チャネルAの入出力信号

$\overline{\text{CTSA}}$(入力)……QDのライトプロテクトの出力"H"のとき書込み禁止,"L"のとき書込み可となります.

$\overline{\text{RTSA}}$(出力)……この信号が"L"のときWRITE GATEは"H"となりQDに対して書込み可とし,$\overline{\text{TXCA}}$への送信用クロック出力を可能とします.

$\overline{\text{DCDA}}$(入力)……QDのメディアSWの出力"L"のときメディアがセットされていることを示します.

$\overline{\text{TXDA}}$(出力)……送信データ出力

$\overline{\text{TXCA}}$(入力)……送信用クロック(TXDA—送信データ—はこのクロックの立下がりでもってサンプリングされ出力されます.), 101.5625 kHz

RXDA(入力)……受信データ入力

RXCA(入力)……受信用クロック(このクロックの立上がりでRXDAのデータをサンプリングし,SIOへ取込みます.)

チャネルBの入出力信号

$\overline{\text{DTRB}}$(出力)……QDへのモータON信号."H"→"L"への立下がりでモータがONされ"H"でQDのモータストップポイントにきたとき,モータOFFされます.

$\overline{\text{DCDB}}$(入力)……QDからの$\overline{\text{HOME}}$信号,この信号の立下がりでREAD/WRITEの動作が開始されます.

$\overline{\text{RTSB}}$(出力)……MFM復調回路のVFO動作のON/OFF信号.データ受信時は"L".この信号の立下がりによってMFM復調回路にビット同期をかけることができます.

SIOのAチャネルは,データの送受信を行うと同時にCRCキャラクタ(データのエラーチェック用)の生成および照合を行っています.Bチャネルは QDのコントロール入出力信号としてのみ使用しています.

MFM変調回路はSIOによってシリアル変換されたデータをMFM記録方式に従って変調します.MFM復調回路はMFM変調されたREAD DATAを通常のシリアルデータに復調します.OSCは変調・復調に必要なクロックパルスを発生します.水晶の原発振は6.5MHzです.ここでMFM復調回路をもう少し詳しく説明しましょう.図3・22がその具体的な回路図です.QDからのREAD DATA出力(RDDT)信号はMFM変調データであり,なおかつインターフェイス回路のクロック(CK…3.25MHz)に対し非同期な信号ですので,Q_0, Q_1, Q_2の各F/FによってCKと同期をとってやります.そしてカウンタ方式のデータセパレータによってデータクロック(RXCA)とデータ(RXDA)に分離し,RXCAの立上がりでRXDAをサンプリングすることにより,SIOはシリアルデータとして受信します.RXCAを正常に動作させるためには,

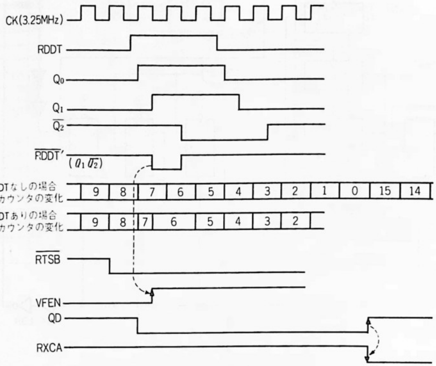


図 3・23
タイミングチャート
（CK 32 パルス分でちょうど
1 ビットセルになる）

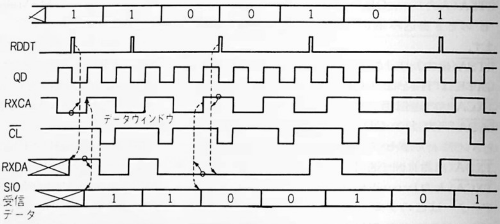


図 3・24
タイミングチャート

ビット同期という作業が必要です．MFM 方式では，連続した“0”の場合，READ DATA はかならずビットセルの先頭にきます．これを利用して QA, QB, QC, QD で構成される 4bit のカウンタを初期設定してやります．“0”のデータが連続しているエリア（GAP 部分）で $\overline{\text{RTSB}}$ を“H”から“L”とします．“L”となってから最初の $\overline{\text{RDDT'}}$（RDDT を CK に同期させた信号）によって VFEN 信号が“H”となり，RXCA が能動状態となります．（VFEN が“L”のとき RXCA は無条件に“H”状態を保持しますが，VFEN が“H”となると QD 信号が立上がるたびに RXCA は反転していきます．）同時に $\overline{\text{RDDT'}}$ が“L”の期間，カウンタは 6（QA＝1, QB＝2, QC＝4, QD＝8 とする）の値となるので QD は“L”状態となります．そして CK によりこのカウンタは 6 からカウントダウンしていき（0 の次は 15 となる）$\overline{\text{RDDT'}}$ がくるまでカウントダウンを繰返します．

いったんこのようにビット同期がとれれば QD と RXCA 関係は READ DATA“1”（ビットセルの中央）が入力されてきても変りません．以上の様子を図 3・23, 3・24 のタイミングチャートに示しておきます．

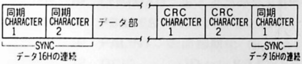
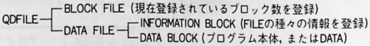


図3・25 QDファイルの基本的なフォーマット



ファイルブロックの種類

補説　RXCAが"L"の期間がデータウィンドウとなります.この期間中にRDDTのパルスがあれば"1"と判定しRXDAを"H"とします. RXDAはビットセルの前半1/4の期間$\overline{CL}$によってセットされるのでデータが"1"の場合のみ"H"となります.

SIOはRXCAの立上がりでRXDAをサンプリングしますので結果としてSIOの受信データは図の通りとなります.

③ピークシフト

RDDT信号がMFM記録方式が定める正規の位置よりずれる現象を, ピークシフトといいます. ピークシフト量が大きくなるとリードミスを発生します. ピークシフト発生の要因は回転数変動, 近接する磁化反転位置間の相互干渉, ヘッド系を含めたリードアンプ部の分解能(f特含む)などが主にあげられます.QDで許容されるピークシフト量は±25%(±2.46μs…ビットセルに対して)です. これ以内のRDDTの変動に対してMFM復調回路は追従補正を行うことができます.

3. QDのファイルフォーマット

(1)ファイルの基本的なフォーマット

CPU側とQD間のデータのREAD／WRITEはZ80SIO(Serial Input／Output)を介して行われていることはすでに述べました.Z80SIOのデータ転送方式としてバイシンク(16bit同期)モードを採用しています. 従ってQDのファイルの基本的なフォーマットは図3・25のようになります.

(2)ファイルブロックの種類

ブロックの種類により, 上のようにデータ部のフォーマットが異なります. ここではそのデータ部のフォーマットについて説明していきます.

①ブロックファイル

このブロックは, 複数ファイルの管理を行うための重要なブロックです. データ部のフォーマットはBLOCKSと称する1byteより構成されており, 現在登録されているブロック数(ブロック・ファイル自身は除外した数)が記録されます(図3・26).

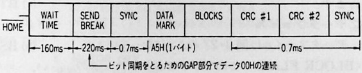


図3・26
ブロックファイル

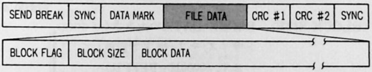


図3・27
データファイル

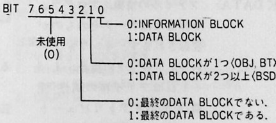


図3・28
ブロックフラグ

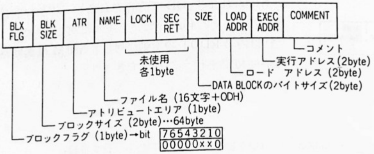


図3・29
インフォメーションブロック

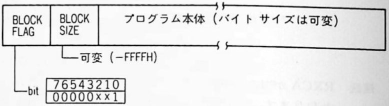


図3・30 A
BTX, OBJ ファイル

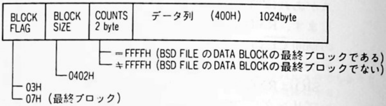


図3・30 B
BSD ファイル

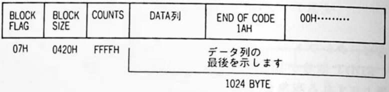


図3・30 C
データブロックの最終ブロック

データマークは同期キャラクタ直後に書かれたA5Hという1バイトのデータで,このデータにより同期キャラクタとデータとを分離します. データマーク以降CRCまでの間に16Hがきても同期キャラクタとしてではなく1データとして処理されます.

②データファイル

データファイルは図3・27のようになっています.

〈BLOCK FLAG〉 このフラグによりブロックの種類を判断します(図3・28).

〈BLOCK SIZE〉 BLOCK DATAの大きさ(バイトサイズ)を2バイトで表現しています.(0000H〜FFFFH)

〈BLOCK DATA〉 ファイルの情報, プログラム本体またはデータそのものが登録されます. インフォメーションブロックの場合, 1バイト目はファイルの属性(種類)を表わすアトリビュートエリアとして使用しています.

(3)具体的なQDのフォーマット

前述のBLOCK DATAのフォーマットを以下のように定めています.

①インフォメーションブロック(図3・29)

②データブロック

(a) BTX, OBJ ファイルは図3・30Aを参照して下さい.

(b) BSD ファイルは図3・30Bを参照してください.

BSD ファイルの, データブロックの最終ブロックは図3・30Cの通りになります.

3-2-5

QDハンドラー

ここでは機械語でQuick diskをハンドリングするためのサブルーチンの説明をします.

モニタ9Z-502Mには次のようなJUMPTABLEがあります.

E800H NOP

E801H JP START : IPL スタートルーチン
E804H JP COMMAD : モニタコマンド待ち
E807H JP PSGRES : PSG の音をとめる
E80AH JP PALINT : 画面初期化
E80DH JP QDIO : QD ハンドラエントリ

QD を扱うためには種々のパラメータを設定し QDIO(E80DH)を CALL します.

パラメータ設定条件を QDIO パラメーター一覧表に示します. モニタ 9Z-502 Mは, QDTBL 以下の WORK は, リスト 3・1 で示すアドレスにて使用し

1 READY CHECK
[機能] QDのReady及びWrite Protect Checkを行なう.
[入力パラメータ]
〈QDTBL〉
QDPA (コマンド) ……1
QDPB (パラメータ) ……=0 Ready Check
≠0 Ready and Write Protect Check
QDPC (Head Addrs 1)……don't care
QDPE (Byte Size 1) ……don't care
QDPG (Head Addrs 2)……don't care
QDPI (Byte Size 2) ……don't care
[出力パラメータ]
CF=0 通常終了
CF=1 エラー
Acc－－－Error Code

2 FORMAT
[機能] QDのReady Checkを行い, 正常終了であれば, Formatを行なう.
[入力パラメータ]
〈QDTBL〉
QDPA (コマンド) ……2
QDPB (パラメータ) ……don't care
QDPC (Head Addrs 1)……don't care
QDPE (Byte Size 1) ……don't care
QDPG (Head Addrs 2)……don't care
QDPI (Byte Size 2) ……don't care
[出力パラメータ]
CF=0 通常終了
CF=1 エラー
Acc－－－Error Code

3 READ
[機能] QDの現在のヘッドポイントより, 指示された Data Fileを検索し, Readする.
[入力パラメータ]
〈QDTBL〉
QDPA (コマンド) ……3
QDPB (パラメータ) ……bit 0 =0 Information read
=1 Data read
bit 1～7 don't care
QDPC (Head Addrs 1)……Information or Data Head Addrs
QDPE (Byte Size 1) ……Information or Data Byte Size
QDPG (Head Addrs 2)……don't care
QDPI (Byte Size 2) ……don't care
[出力パラメータ]
CF=0 通常終了
CF=1 エラー
Acc－－－Error Code

4 WRITE
[機能] QDのData Write Endを検出し 指定のDataをWriteする.
[入力パラメータ]
〈QDTBL〉
QDPA (コマンド) ……4
QDPB (パラメータ) …bit 0 =0 Information+Data write
=1 Data write
bit 1 =0 No Block Data (ex. OBJ BTX)
=1 Block Data (ex BSD)
bit 2 =0 Not Close
=1 Close
bit 3～7 don't care
QDPC (Head Addrs 1)……Information Head Addrs
QDPE (Byte Size 1) ……Information Byte Size
QDPG (Head Addrs 2)……Data Head Addrs
QDPI (Byte Size 2) ……Data Byte Size
[出力パラメータ]
CF=0 通常終了
CF=1 エラー
Acc－－－Error Code

5 HEADER POINT CLEAR
[機能] QDの現在のヘッドポイントをクリアする.
[入力パラメータ]
〈QDTBL〉
QDPA (コマンド) ……5
QDPB (パラメータ) ……don't care
QDPC (Head Addrs 1)……don't care
QDPE (Byte Size 1) ……don't care
QDPG (Head Addrs 2)……don't care
QDPI (Byte Size 2) ……don't care
[出力パラメータ]
なし
[注意] 次の場合, このコマンドを実行する
その他の場合は, IOCSが管理
1. Error40(File not found) が発生した場合
2. OBJ BTXファイルのRead後及びROPEN後のCLOSE時
3. DIR終了時

6 MOTOR OFF
[機能] QDのMotor off
[入力パラメータ]
〈QDTBL〉
QDPA (コマンド) ……6
QDPB (パラメータ) ……don't care
QDPC (Head Addrs 1)……don't care
QDPE (Byte Size 1) ……don't care
QDPG (Head Addrs 2)……don't care
QDPI (Byte Size 2) ……don't care
[出力パラメータ]
なし
[注意] 次の場合, このコマンドを実行する
その他の場合は, IOCSが管理
1. Save時Error42または, Error51 (Already exist error, Too many file error) が発生した場合
2. DIR終了時

QDIO のパラメータ一覧

```
**  Z80 ASSEMBLER 2Z-004C  <SAMPLE>  PAGE 01                08.21.84

0000              ;
0000              ;================================
0000              ;
0000              ;        EQUATE  FILE
0000              ;
0000              ;
0000              ;================================
0000              ;
E80D   P          QDIO:    EQU     E80DH          ;QD IOCS ENTRY
0000              ;
0000              ; QUICK DISK WORK TABLE
0000              ;
1130   P          QDTBL:   EQU     1130H          ;IOCS PARAMETER ADRS
1130   P          QDPA:    EQU     QDTBL          ;IOCS COMMAND
1131   P          QDPB:    EQU     QDPA+1         ;IOCS PARAMETER
1132   P          QDPC:    EQU     QDPB+1         ;DATA HEAD ADRS(1)
1134   P          QDPE:    EQU     QDPC+2         ;DATA BYTE SIZE(1)
1136   P          QDPG:    EQU     QDPE+2         ;DATA HEAD ADRS(2)
1138   P          QDPI:    EQU     QDPG+2         ;DATA BYTE SIZE(2)
0000              ;
113C   P          QDCPC:   EQU     QDPI+4         ;QD FLAG BYTE
113F   P          FNUPS:   EQU     QDCPC+3
1141   P          FNUPF:   EQU     FNUPS+2
1144   P          MTF:     EQU     FNUPF+3
0000              ;
0000              ;     QD INFORMATION
0000              ;
11F0   P          QATRB:   EQU     11F0H          ;QD ATRB
11F0   P          QNAME:   EQU     11F0H          ;QD NAME
1104   P          QSIZE:   EQU     1104H          ;QD FILE SIZE
1106   P          QDTADR:  EQU     1106H          ;QD DATA ADDR
1108   P          QEXADR:  EQU     1108H          ;QD EXEC ADDR
0000              ;
0000              ;
0011   P          NAMSIZ:  EQU     0011H          ;NAME SIZE
11A3   P          BUFF:    EQU     11A3H          ;WORK  BUFFER
0000              ;
0000              ;      QD ERR   CODE
0000              ;
0000              ; ERR 40;NOT FOUND ERR
0000              ; ERR 41;HARD ERR
0000              ; ERR 42;ALREADY EXIST
0000              ; ERR 46;WRITE PROTECT ERR
0000              ; ERR 50;NOT READY
0000              ; ERR 51;TOO MANY FILES ERR
0000              ; ERR 53;NO FILE SPACE ERR
0000              ; ERR 54;UNFORMAT ERR
0000              ; ERR 57;BAD DISK ERR
0000              ;
0000                       SKP     H
```

リスト3・1 QDハンドラー

ています．また Error Code は BASIC MZ5Z-001 と同一のコードです(リスト中に QDIO で発生するエラーコードを示してあります)．

次に QDIO の使用例を示します(リスト3・1参照)．

例1　ファイルの LOAD

QLOAD で始まるサブルーチンは，Quick disk で最初に見つけられたファイルをそのファイルの Information に従って LOAD します．エラー発生時は，キャリーフラグが立って，もどってきます．まず IOFRS にて，IOCS で使用する WORK を初期化し，メディアが Read 可能な状態かどうか，QDRCK にて Ready Check を行います．

次に QRDI で最初に見つけられた Information BLOCK を読出します． Information BLOCK にはアトリビュート，ファイル名，サイズ，LOAD アドレス，実行アドレス等の情報が含まれています．これらの情報に基づき DATA BLOCK を読出します．

名　前	バイト数	機　　能
QATRB	1	属　性
QNAME	17	ファイル名(16文字まで，最後ODH)
LOCK	1	使用していない（予備）
SECRET	1	〃
QSIZE	2	ファイルのバイト数(下位，上位の順)
QDTADR	2	LOADINGするアドレス
QEXADR	2	実行するアドレス
COMENT	38	コメント

表3・11　インフォメーションブロックの内容

例2　ファイルの SAVE

QSAVE で始まるサブルーチンは Quick disk に QATRB 以下の設定に基づき File をセーブします．エラー発生時には，キャリーフラグが立ってもどってきます(表3・11参照)．

```
**  Z80 ASSEMBLER 2Z-004C  <SAMPLE>  PAGE 02          08.21.84

0000                  ;==========================
0000                  ;
0000                  ;   QUICK DISK LOAD
0000                  ;
0000                  ;==========================
0000                  ;
0000                  QLOAD:
0000 CDC200                   CALL  IOFRS          ;IOCS FLAG RESET
0003 CDB200                   CALL  QDRCK          ;READY CHECK
0006 DAB000                   JP    C,QERROR
0009                  ;
0009 CDCD00                   CALL  HDPCL          ;HEAD POINT CLEAR
000C                  ;
000C CD2700                   CALL  QRDI           ;READ INFORMATION BLOCK
000F DAB000                   JP    C,QERROR
0012                  ;
0012                  ; READ DATA BLOCK
0012                  ;
0012 2A0611                   LD    HL,(QDTADR)
0015 223211                   LD    (QDPC),HL      ;DATA ADDR SET
0018 2A0411                   LD    HL,(QSIZE)
001B 223411                   LD    (QDPE),HL
001E 210301                   LD    HL,0103H       ;READ DATA BLOCK CMD.
0021 223011                   LD    (QDPA),HL
0024 C30DE8                   JP    QDIO           ;QD IOCS
0027                  ;
0027                  ; READ INFORMATION BLOCK
0027                  ;
0027                  QRDI:
0027 210300                   LD    HL,0003H       ;READ INFORMATION CMD.
002A 223011                   LD    (QDPA),HL
002D 21F011                   LD    HL,QATRB       ;HEAD ADDR
0030 223211                   LD    (QDPC),HL
0033 214000                   LD    HL,0040H       ;READ SIZE
0036 223411                   LD    (QDPE),HL
0039 C30DE8                   JP    QDIO
003C                  ;
003C                  ;
003C                          SKP   H
```

リスト3・1　QDハンドラー

```
**  Z80 ASSEMBLER 2Z-004C <SAMPLE>  PAGE 03                  08.21.84

003C                    ;==============================
003C                    ;
003C                    ;   QUICK DISK SAVE
003C                    ;
003C                    ;==============================
003C                    ;
003C                    QSAVE:
003C CDC200                     CALL   IOFRS          ;IOCS FLAG RESET
003F CDB500                     CALL   QDRWCK         ;READY & WPRT CHECK
0042 DAB000                     JP     C,QERROR
0045                    ;
0045                    ;
0045                    ; FILE END SEARCH
0045                    ;
0045 CD7000                     CALL   FEDSCH
0048 DAB000                     JP     C,QERROR
004B                    ;
004B                    ; SAVE FILE START
004B                    ;
004B FE28               SVFLST: CP     40             ;NOT FOUND ?
004D 2061                       JR     NZ,QERROR
004F                    ;
004F                    ; WRITE INF+DATA BLOCK
004F                    ;
004F 2A0611                     LD     HL,(QDTADR)    ;DATA TOP ADDR
0052 223611                     LD     (QDPG),HL
0055 210404                     LD     HL,0404H       ;WRITE INF+DATA BLK CMD.
0058 223011                     LD     (QDPA),HL
005B 21F011                     LD     HL,QATRB       ;INFM TOP ADDR
005E 223211                     LD     (QDPC),HL
0061 214000                     LD     HL,0040H       ;INFM BYTE SIZE
0064 223411                     LD     (QDPE),HL
0067 2A0411                     LD     HL,(QSIZE)     ;DATA BYTE SIZE
006A 223811                     LD     (QDPI),HL
006D C30DE8                     JP     QDIO
0070                    ;
0070                    ;
0070                    ; FILE END SEARCH SUB.
0070                    ;
0070                    FEDSCH:
0070 CDCD00                     CALL   HDPCL          ;HEAD POINT CLEAR
0073                    ;
0073 210300                     LD     HL,0003H       ;READ INFORMATION CMD.
0076 223011                     LD     (QDPA),HL
0079 21A311                     LD     HL,BUFF        ;HEAD ADDR
007C 223211                     LD     (QDPC),HL
007F 214000                     LD     HL,0040H       ;READ SIZE
0082 223411                     LD     (QDPE),HL
0085                    ;
0085                    ; TOO MANY FILES ?
0085                    ;
0085 AF                         XOR    A
0086 323C11                     LD     (QDCPC),A
0089 3A3C11             EDFLCK: LD     A,(QDCPC)
008C 3C                         INC    A
008D FE21                       CP     33             ;FILES
008F 323C11                     LD     (QDCPC),A
0092 3E33                       LD     A,51           ;TOO MANY FILES ERR
0094 D0                         RET    NC
0095                    ;
```

リスト3・1 QDハンドラー

```
 **   Z80 ASSEMBLER 2Z-004C  <SAMPLE>  PAGE 04            08.21.84

0095                          ; END FILE CHECK
0095                          ;
0095 CD0DE8                            CALL   QDIO
0098 3F                                CCF
0099 D0                                RET    NC
009A                          ;
009A                          ; SAME FILE NAME CHECK
009A                          ;
009A 11A411                            LD     DE,BUFF+1
009D 21F011                            LD     HL,QNAME
00A0 0611                              LD     B,11H
00A2 1A                       SFNC:    LD     A,(DE)
00A3 BE                                CP     (HL)
00A4 20E3                              JR     NZ,EDFLCK
00A6 FE0D                              CP     0DH
00A8 2804                              JR     Z,SFNC0
00AA 13                                INC    DE
00AB 23                                INC    HL
00AC 10F4                              DJNZ   SFNC
00AE 3E2A                     SFNC0:   LD     A,42             ;ALREADY EXIST ERROR
00B0 37                       QERROR:  SCF                      ;CY=1
00B1 C9                                RET
00B2                          ;
00B2                          ;================================
00B2                          ;
00B2                          ;        QD UTILITY SUB.
00B2                          ;
00B2                          ;================================
00B2                          ;
00B2                          ; READY & WRITE PROTECT CHECK
00B2                          ;
00B2 AF                       QDRCK:   XOR    A
00B3 1802                              JR     QDRCK1
00B5                          ;
00B5 3EFF                     QDRWCK:  LD     A,FFH
00B7 323111                   QDRCK1:  LD     (QDPB),A
00BA 3E01                              LD     A,01H
00BC 323011                            LD     (QDPA),A
00BF C30DE8                            JP     QDIO
00C2                          ;
00C2                          ;
00C2                          ; IOCS FLAG RESET
00C2                          ;
00C2                          IOFRS:
00C2 AF                                XOR    A                ;Acc←00H
00C3 324411                            LD     (MTF),A          ;MOTOR FLAG
00C6 323F11                            LD     (FNUPS),A        ;FILE NUMBER FLAG
00C9 324111                            LD     (FNUPF),A        ;FILE NUMBER UP FLAG
00CC C9                                RET
00CD                          ;
00CD                          ; HEAD PONITER CLEAR
00CD                          ;
00CD 3E05                     HDPCL:   LD     A,5
00CF 323011                            LD     (QDPA),A
00D2 C30DE8                            JP     QDIO
00D5                          ;
00D5                                   END

 **   Z80 ASSEMBLER 2Z-004C  <SAMPLE>  PAGE 05            08.21.84

BUFF    11A3  EDFLCK 0089  FEDSCH 0070  FNUPF  1141  FNUPS  113F
HDPCL   00CD  IOFRS  00C2  MTF    1144  NAMSIZ 0011  QATRB  11F0
QDCPC   113C  QDIO   E80D  QDPA   1130  QDPB   1131  QDPC   1132
QDPE    1134  QDPG   1136  QDPI   1138  QDRCK  00B2  QDRCK1 00B7
QDRWCK  00B5  QDTADR 1106  QDTBL  1130  QERROR 00B0  QEXADR 1108
QLOAD   0000  QNAME  11F0  QRDI   0027  QSAVE  003C  QSIZE  1104
SFNC    00A2  SFNC0  00AE  SVFLST 004B
```

リスト3・1 QDハンドラー

3-2-6 カセットインターフェイス

カセットインタフェース

(1)記録の仕組み

MZ-1500は，PWM(Pulse Width Modulation)という，パルス幅を変えることによりデータを記録する方式を採用しており，高い信頼度を確保しています．

PWM方式では“1”のパルス幅を“0”の倍にしており，ヘッドの出力波形とパルスの位相関係は図3・31のようになります．“1”と“0”のパルス幅が異なるため，データの転送速度は一定ではなくデータのパターンにより変化し，同じバイト数でも“0”だけのデータは“1”だけのデータの2分の1の時間で読み書きできることになります．

次にPWM方式により記録された“0”，“1”の信号をCPUがどのようにして見分けるのか説明します．図3・32のように，CPUは読出された信号が“Low”→“High”になる「立上がりエッジ」を待ち，見つけた時点からある一定時間(a)後の信号が“Low”であるか，“High”であるかにより，“0”，“1”を見分けます．この動作を8回くり返し，1バイトになるとこのデータをメモリに書込みます．

(2)信号発生のメカニズムとソフトウェア

信号発生は8255を使用して行ないます．入出力ポートCの下位4ビットは出力モードに設定されており，このうちのPC_1がデータレコーダのWrite端子に接続されています．8255にはポートCの出力モードになっているビットに対してコントロールワードを書込むことによりセット，リセットする機能があるのでこれを利用し，図3・33のシーケンスにて“0”，“1”の信号を発生することができます．

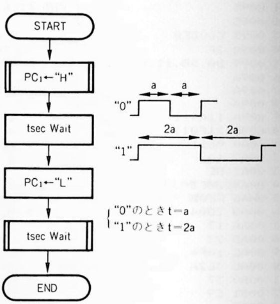


図3・33　信号発生のシーケンス

データレコーダ側が何もSWを押していない状態ではSENCE信号は“H”になっており，SW(REW，FF，PLAY)を押すと“L”になります．SENCE信号が“H”→“L”となると，D-FFにプリセットがかかり，MOTOR信号を“H”としてモータが回転します．

ここで補足になりますが，データレコーダのモータコントロールについて説明します．モータのコントロールにも8255を利用しており，図3・34のような回路になっています．

また，データレコーダ側でREW，FF，PLAYのいずれかのSWが押されている場合，PC_3にてD-

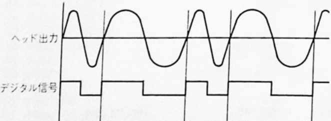


図3・31　PWM方式

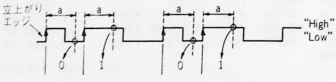


図3・32　PWMにおける信号の識別

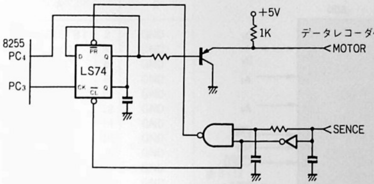


図3・34
モータコントロール回路

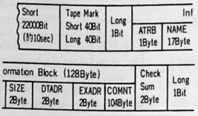
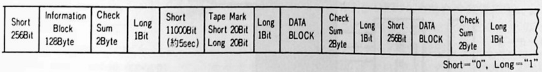



図3・35 データフォーマット

FFにクロックを与えることによりモータのオン，オフをコントロールすることができ，PC_4を用いてモータの状態を調べることもできます．

(3)各種の情報

カセットテープのデータフォーマットは図3・35のようになっています．

パソコンが"0"，"1"の列を5バイトずつのデータとして読込むためには，まず，バイト同期をかける必要があります．図3・35のフォーマットでは「Tape Mark」でバイト同期をかけ，その次のデータを読込むこととなります．下記にインフォメーションブロックの内容を示します．

名前　　機　　　能

ATRB　：ファイルの属性

NAME　：ファイル名（16文字＋0D）

SIZE　：データブロックのバイトサイズ

DTADR：データをローディングするアドレス

EXADR：ローディング後の実行アドレス

COMNT：コメント

このインフォメーションブロックにて得られた情報にしたがって，データブロックを読込み，実行します．

また信頼度をより高くするため，インフォメーションブロック，データブロックとも2回ずつ記録されています．

3-2-7 キーボードスキャン

MZ-1500のキーボードスキャンニングはメインCPU(Z80CPU)による，ソフトウェアスキャンニングを行っており，スピードの早いキー入力に対処するように配慮されています．キーボードスキャンには，汎用の入出力ICである8255を使っています．8255はキーボードコントロールの他に，カセットレコーダのコントロールにも使用しています．

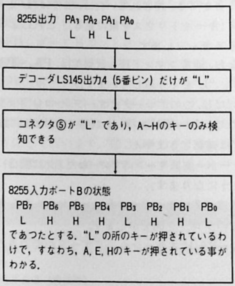


図3・36 キーボードスキャンのフロー

キーを検知するには，PA_0～PA_3を用いて，キーストローブを出力します．

これがデコーダに入力され，デコーダは出力0～9のうちいずれかだけを"L"とします．

これが，キーマトリクスに加えられ，この"L"が入力しているキーのグループ(キーマトリクスを縦に見る)のみ，キーが押されているかどうかを調べ

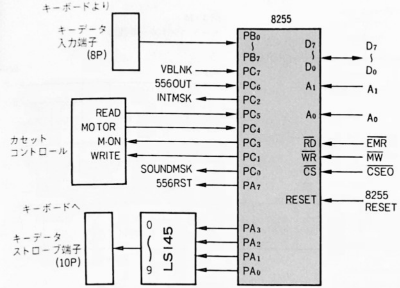


図3・37
8255周辺のブロック図

ることができ，押されていれば，信号は“L”となります(キーマトリクスを横に見る)．これにより，キーをスキャンします．

なお，通常コマンド待ち状態では，PA_0～PA_3は“H”，“L”をくり返しています．

ただし，このデコーダはオープン・コレクタであるので，キーSWを押した状態でなければ“H”，“L”は確認できません．

キーボードスキャンのフローはたとえば図3・36のようになります．

図3・37にキーボードをコントロールする8255周辺のブロック図を示します．

PA_0～PA_3にスキャンニングデータが出力され，LS145のデコーダによって10本のストローブがキーのストローブに出力されます．

図3・38に示すキーマトリクスを通して(キーマトリクスは，押しボタンスイッチの集まりと考えてください)，キーのデータが8255のPB_0～PB_7に入ります．

3-2-8
プリンタインターフェイス

(1)信号のやりとりの原理，信号の情報

コンピュータを用いてプリンタに文字を印字させるためには8ビットのデータをプリンタに送る

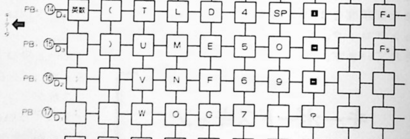


図3・38
キーマトリクス

	A		B
1	RDP($\overline{STB}$)	2	GND
3	RD1	4	GND
5	RD2	6	GND
7	RD3	8	GND
9	RD4	10	GND
11	RD5	12	GND
13	RD6	14	GND
15	RD7	16	GND
17	RD8	18	GND
19	IRT($\overline{RST}$)	20	GND
21	RDA(BUSY)	22	GND
23	STA	24	GND

表3・12
プリンタ端子と
信号配列

ことが必要です．しかしながら，コンピュータとプリンタのデータの信号線を接続して，次々とデータを送るだけでは印字はできません．というのは，コンピュータはプリンタが今データを受け入れることが可能な状態かどうかを知ることができないからです．またプリンタは印字データをいつ受け取ればいいのかもわかりません．そこで8ビットの印字データを送る信号の他に，プリンタのデータ受入れ状態を示す信号(プリンタ→コンピュータ)，プリンタにデータの受取りを指示する信号(コンピュータ→プリンタ)が必要になります．基本的にはこれだけの信号によってプリンタに印字することができます．(他に信号を用いているインターフェイスもありますが基本的には上記のものと考え方は同じです．)

次にMZ-1500のプリンタインターフェイスについて説明していきます．

①信号配置および説明

MZ-1500のプリンタインターフェイスの信号はMZ-1500後面の外部プリンタ端子に，表3・12のような信号配列で入出力されます．表3・13に各信号線の意味を説明します．ただしカッコ内はセントロニクスインターフェイスに設定(MZ-1500のシステムSW2をOFF)した場合の説明です．

さて，この信号線を用いてどのようにしてプリンタに印字させるのかを説明しましょう．

MZ-1500はプリンタインターフェイスにZ80PIOを使用しています．Z80PIOはAとBの2組(8ビット)の入出力ポートを持っており，各ポートのアドレスは以下の通りです．表3・14も参照ください．

$FC　ポートAコントロール
$FD　ポートBコントロール
$FE　ポートAデータ
$FF　ポートBデータ

Z80PIOのポートにデータを入出力することによって印字するのですが，その具体的方法は以下のようにして行います．(図3・39のタイミングチャートおよび図3・40の印字データ転送ルーチンを

ピン	信号		備考
1	RDP ($\overline{STB}$)	OUT	データの読込みを指示する信号ストローブ信号，信号がhigh (low) レベルとなった後にデータはプリンタに読込まれる 正論理である (負論理)
2 5 7 9 11 13 15 17	RD_0 RD_1 RD_2 RD_3 RD_4 RD_5 RD_6 RD_7	OUT	8ビットパラレルデータの1ビット目～8ビット目までのデータを表わす RD0はLSB, RD7はMSBに対応する
19	IRT ($\overline{RST}$)	OUT	イニシャルリセット信号であり，強制的にプリンタを電源投入状態にもどす 正論理である (負論理)
21	$\overline{RDA}$ (BUSY)	IN	プリンタが動作可能か否か (データのエントリが可能か否か) を示す 負論理 (正論理) であり，信号がlowレベルの状態でデータのエントリが可能となる
23	STA*	IN	印字用紙不足またはメカニズムのトラブル発生時この信号はlowレベルである 正常な状態ではhighレベルである

表3・13
各信号の意味

	端子	I/O	アクティブ	信号名	役　割
ポートA ($FE)	PA_0	IN	L	$\overline{RDA}$	"L"でプリンタが受信可能状態を示す
	PA_1	IN	L	$\overline{STA}$	ステータスチェック　紙なしの時"L"
	PA_2	IN	−	GND	
	PA_3	IN	−	GND	
	PA_4	IN	H	INTO	8253出力で割込みに使用可
	PA_5	IN	H	INTI	8253出力で割込みに使用可
	PA_6	OUT	H	IRT	プリンタを初期化する
	PA_7	OUT	H	RDP	プリンタにデータの受取りを指示する
ポートB ($FF)	PB_0	OUT	−	RD_0	プリンタへの印字データ あるいはコントロールコード
	PB_1	OUT	−	RD_1	
	PB_2	OUT	−	RD_2	
	PB_3	OUT	−	RD_3	
	PB_4	OUT	−	RD_4	
	PB_5	OUT	−	RD_5	
	PB_6	OUT	−	RD_6	
	PB_7	OUT	−	RD_7	

表3・14　ポートA，Bとその役割

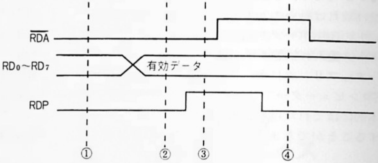


図3・39
タイミングチャート

MZ仕様		セントロニクス仕様	
信号名	アクティブ	信号名	アクティブ
$\overline{RDA}$（注）	"L"	BUSY	"H"
RDP	"H"	$\overline{STB}$	"L"
IRT	"H"	$\overline{RST}$	"L"

表3・15　MZ仕様とセントロニクス仕様の違い

参照して下さい.)

まず，コンピュータ側はプリンタが受け入れ可能状態であることを確認します(①).　次に印字データを$FFポートへ送ります(②).　この後プリンタへデータ受け取り指示をRDPを用いて行ない(③)，プリンタ側がデータを受け取った事を確認($\overline{RDA}$="H"であるのを確認)してからRDPを"L"にします(④).以上で1文字分のデータ転送は"H"となった$\overline{RDA}$信号が"L"になるまで待ってから行います.

またPIOの$\overline{BSTB}$入力に$\overline{RDA}$信号が入力されており，$\overline{RDA}$の立下がりにおいて，CPUに対して割込みをかけて続くデータ転送を行わせることも可能です(ポートBモード0指定時).

以上の方法はMZ仕様のインターフェイスを持ったプリンタに対するインターフェイスの方法ですが,プリンタとのパラレル(8ビット並列データ転送)インターフェイスの方法として,セントロニクス社仕様の方法もあります.先にも述べていますが，この方法とMZの方法は基本的には同じであり，ただ一部の信号がアクティブハイかローかが異なっていると考えられます(表3・15参照).

注)$\overline{RDA}$はアクティブローですが信号の意味としてはプリンタが受信可能状態を"L"としており，

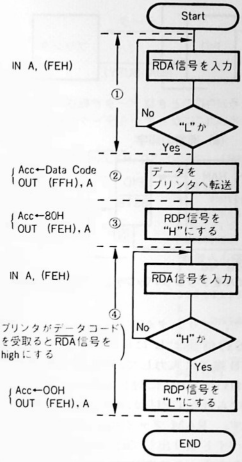
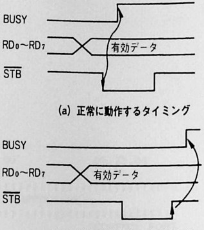


図3・40　印字データ転送ルーチン

BUSYはプリンタが受信不可能状態を"H"としているため表現は異っているが信号としては同じものとして扱えます.

従ってセントロニクス仕様のプリンタを接続するためにはRDPとIRTの信号の極性を反転すればよいことが分かります.

MZ-1500では図3・41のような回路をインターフェイス部分に組入れて，ディップSWによってインターフェイス仕様を切り換えるようにしてあ



(a) 正常に動作するタイミング

(b) 正常に動作しないタイミング

図3・42　セントロニクス仕様での動作タイミング

ります.

しかし，インターフェイスがMZ仕様あるいはセントロニクス仕様であって，ハードウェアとしてはMZ-1500に接続できたとしても，CGパターンやプリンタコントロールコード等のソフトウェアがMZ-1500専用プリンタと異なると正しく動作しない場合があります．また，同じセントロニクス仕様のプリンタでも以下のようなものはハードウェア的に接続できません.

図3・42(a)では$\overline{STB}$の立下がりでBUSY信号が"H"になります．しかし(b)のように$\overline{STB}$の立上がりで，BUSY信号が"H"になるものに対してはうまく動作しません．というのは，$\overline{STB}$はBUSYの"H"状態をチェックしてから"H"状態にするのですが(b)のようなものではいつまでもBUSYが"L"状態であって$\overline{STB}$を"H"にすることができないからです.

また，セントロニクス仕様のプリンタには，今までふれてきた信号の他に，$\overline{ACKNLG}$，$\overline{FAULT}$，PE(ペーパーエンプティ)，SELECT等の情報信号

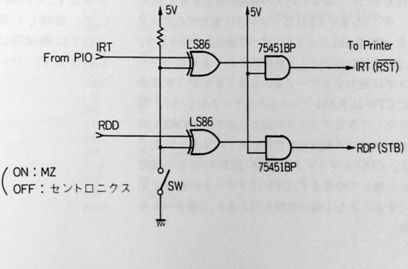
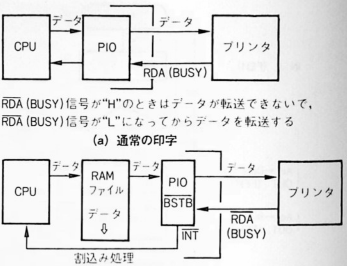


図3・41
IRT，RDP極性
切換え回路例



$\overline{\text{RDA}}$ (BUSY)信号が"H"のときはデータが転送できないで，
$\overline{\text{RDA}}$ (BUSY)信号が"L"になってからデータを転送する

(a) 通常の印字

図3・43
プリンタバッファの考えかた (b) RAMファイルをプリンタバッファとして使用した場合の印字

がありますが，特にMZ-1500とのインターフェイスには使用することはありません．

(2) PIOのアプリケーション

プリンタの動作はCPUの実行に比べるとずいぶん遅いものです．少し長めのプログラムリストをプリントアウトしようものなら，そのプリントアウトの間，コンピュータは他の仕事は何もできなくなります．ずっと$\overline{\text{RDA}}$(BUSY)信号が"L"になるのを待っていなくてはなりません．しかしMZ-1500にはRAMファイルという強い味方がいます．このRAMファイルをプリンタバッファとして使用すると，プリンタの動作中でも他の処理が可能なのです．そこでプリンタバッファについてもう少し詳しく説明してみることにします．

CPUは$\overline{\text{RDA}}$(BUSY)信号が"L"になってから次の処理をするのですが，プリンタに印字データを送ってプリンタが動作している間は$\overline{\text{RDA}}$(BUSY)信号が"H"であるため次の処理が行えません．

そこで，まずCPUはプリンタに出力するデータを一度，RAMファイルにすべて書込んでやります．この実行にはほとんど時間はかかりません．(プリンタに出力するデータ量にもよりますが．)さて次にCPUはRAMファイルからデータを1バイト取り出してきてプリンタに出力します．この時1バイトのデータを出力した後はすぐにリターンをして，CPUをプリンタの$\overline{\text{RDA}}$(BUSY)信号の管理から離してやります．CPUはプリンタの実行を気にすることなく他の処理を行えます．(図3・43参照)．

さて，プリンタが実行を終了したら当然$\overline{\text{RDA}}$(BUSY)信号は"L"に戻ります．この信号をPIOの$\overline{\text{BSTB}}$端子に入力して，$\overline{\text{RDA}}$(BUSY)の立ち下がりにおいてCPUに対し割込みをかけます．割込みをかけられたCPUは，割込みルーチンへジャンプします． RAMファイルから割込みルーチンはまた1バイト取り出してきてデータを出力することになります．この割込みルーチンは非常に簡単なものですので，割込み処理による他のプログラムの実行にはさほど大きな影響は与えません．このような処理はポートBモード0指定時に可能になります．

(3)その他

今までプリンタインターフェイスとしてMZ-1500の8ビットのパラレルインターフェイスを紹介してきましたが，このプリンタ出力端子を利用してさまざまな周辺機器を接続することも可能になってきます．その場合には$\overline{\text{STA}}$信号線も大いに活用して，素晴しい周辺装置を考えてみて下さい．

以下に機械語によるプリンタルーチンを紹介しておきます．せいぜい利用してみてください．

```
**  Z80 ASSEMBLER 2Z-004C  <PRINT>  PAGE 01                09.25.84

0000                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0000                    ;     PRINTER CONTROL SUBROUTINE
0000                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
00FF     P              PORT/I: EQU     FFH                 ;CONTROL PORT
00FE     P              PORT/O: EQU     FEH                 ;DATA    PORT
0000                    ;
0000                    ; JP ABNML: NO POWER,NO PAPER OR MECHANICAL TROUBLE
0000                    ;
0000                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0000                    ;
0000                    ;    DATA CODE TRANSFER SUBROUTINE
0000                    ;
0000                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0000                    ;
0000 F5                 PRINT:  PUSH    AF
0001 3E00                       LD      A,00H
0003 CD1600                     CALL    RDA/CK              ; RDA SIGNAL IS LOW?
0006 F1                         POP     AF
0007 D3FE                       OUT     (PORT/O),A          ; DATA TRANSFER
0009 3E80                       LD      A,80H
000B D3FF                       OUT     (PORT/I),A          ; IT MAKES RDP HIGH
000D 3E01                       LD      A,01H
000F CD1600                     CALL    RDA/CK              ; RDA SIGNAL IS HIGH?
0012 AF                         XOR     A
0013 D3FF                       OUT     (PORT/I),A          ; IT MAKES RDP LOW
0015 C9                         RET
0016                    ;
0016                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0016                    ;     RDA SIGNAL CHECK SUBROUTINE
0016                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0016                    ;
0016 C5                 RDA/CK: PUSH    BC
0017 D5                         PUSH    DE
0018 57                         LD      D,A
0019 1E0C                       LD      E,0CH
001B 010000                     LD      BC,0000H            ; FOR TIME DELAY
001E DBFF               LOOP:   IN      A,(PORT/I)
0020 E60D                       AND     0DH
0022 BA                         CP      D                   ; CHECK RDA SIGNAL
0023 2003                       JR      NZ,+5
0025 D1                         POP     DE
0026 C1                         POP     BC
0027 C9                         RET                         ; NORMAL RETURN
0028 0B                         DEC     BC
0029 78                         LD      A,B
002A B1                         OR      C
002B 20F1                       JR      NZ,LOOP
002D 1D                         DEC     E
002E 20EE                       JR      NZ,LOOP
0030 D1                         POP     DE
0031 C1                         POP     BC
0032 C30000        E            JP      ABNML
0035                            END
```

1バイトのデータ転送プログラム

```
**   Z80 ASSEMBLER 2Z-004C  <RESET>   PAGE 01                  09.25.84

0000                          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0000                          ;          PRINTER RESET SUBROUTINE
0000                          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0000                          ;
00FE   P                      PORT/I: EQU      FEH                ; CONTROL PORT
00FF   P                      PORT/O: EQU      FFH                ; DATA    PORT
0000                          ;
0000 F5                       RESET:  PUSH     AF
0001 C5                               PUSH     BC
0002 3E40                             LD       A,40H
0004 D3FE                             OUT      (PORT/I),A         ; IT MAKES IRT HIGH
0006 0B                       LOOP:   DEC      BC
0007 78                               LD       A,B
0008 B1                               OR       C
0009 20FB                             JR       NZ,LOOP            ; TIME DELAY
000B AF                               XOR      A
000C D3FE                             OUT      (PORT/I),A         ;IT MAKES IRT LOW
000E C1                               POP      BC
000F F1                               POP      AF
0010 C9                               RET
0011                                  END
```

プリンタをソフトウェアでリセットするプログラム

3-2-9
PSG

(1) MZ-1500での使われ方

MZ-1500では、TI社(テキサス　インスツルメンツ社)の，SN76489ANというICを2個使用して，これにより音を発生させることができるようになっています．TI社ではこのICをSGC(Sound Generation Controller)と名付けていますが一般的にこのようなICはPSG(Programmable Sound Generator)と呼ばれています．

このICの特徴として，3個のトーン発生器，1個のノイズ発生器を持っており，各発生器ごとに減衰器を持っています．そしてその発生周波数および減衰量は独自にプログラム可能となっています．それとは別に外部よりの入力端子を持っていますので，合計3＋1＋1の信号がミキシングされて，最終的にPSGの出力となります．

MZ-1500では，この2個のPSGの出力を，ステレオアンプ等に接続することにより2つのスピーカを同時に鳴らすことができるようになっています．ただし，MZ-1500の内蔵スピーカは1個だけですので，2個のPSGの出力をさらにミキシングして増幅し，内蔵スピーカを鳴らしています（図3・44参照）．

それぞれのPSGの外部入力端子には8253のSOUND出力(ただし，8255のPC 0を“L”にすることによりマスクすることができる)および，オプションのボイスボードの出力がつながるようになっています．PSGのコントロールは，次のポートへデータを出力することで行いますが，2つのPSGへ同時に同じデータを書込むこともできます．

(2) PSGの構成

PSGの内部ブロックは図3・45のようになっています．

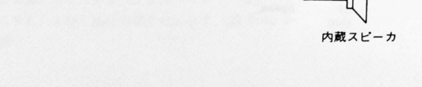


図3・44
MZ-1500での使われ方

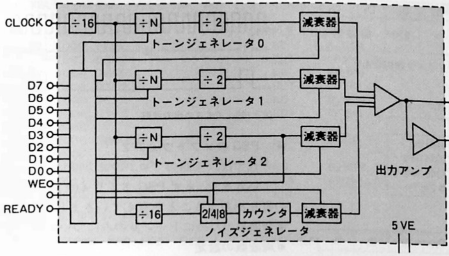


図3・45
PSGの内部構成

第1バイト

D7	D6	D5	D4	D3	D2	D1	D0
1	レジスタ			周波数1			
	R2	R1	R0				

第2バイト

D7	D6	D5	D4	D3	D2	D1	D0
0	×	周波数2					

周波数(f) $= \frac{3.58\times10^3}{32\times n}$ (kHz)　　分周比 $= \frac{3.58\times10^6}{32\times f}$

ただし，nは上位6ビットを周波数2，下位4ビットを周波数1とする10ビットデータ．

表3・16
周波数の設定

3個のトーンジェネレータはそれぞれ，周波数設定部（分周比設定），およびレベル設定部（アッテネーションコントロール）より構成されます．各々のトーンジェネレータは周波数設定データに10 bit，アッテネーション設定データに4 bitを必要とします．また，ノイズジェネレータは，ノイズの種類の選択に1 bit，ノイズ周波数設定データに2 bit計3 bitのデータが必要となります．これと同時に8個の内部レジスタを指定するデータが3 bit必要となりますので，周波数の設定は2バイト転送，アッテネーションおよびノイズコントロールは1バイト転送でデータを出力してやる必要があります．

(3)　PSGのソフトウェア1

a）　周波数の設定

周波数は表3・16の2バイトのデータで設定します．

nは10 bitなので，10進では1～1023までとなります．したがって出力周波数は

最大　$3.58\times10^6/(32\times1) = 111{,}875$ Hz
（実際には聞こえません）

最小　$3.58\times10^6/(32\times1023) \fallingdotseq 109$ Hz

となります．

なおレジスタの指定は表3・17の3 bitで行います．

b）　ノイズの設定

ノイズは，同期ノイズまたはホワイトノイズを発生させることができます．1バイトデータで設定を行います（表3・18）．

ノイズの出力については図3・46を参照して下さい．

●ホワイトノイズを指定したときの出力波形はランダム波形となり，ホワイトノイズに近いスペクトル分布を示す音が出力されます．

R2	R1	R0	コントロールレジスタの割当て
0	0	0	トーン0　周波数
0	0	1	トーン0　音量
0	1	0	トーン1　周波数
0	1	1	トーン1　音量
1	0	0	トーン2　周波数
1	0	1	トーン2　音量
1	1	0	ノイズコントロール
1	1	1	ノイズ音量

表3・17　レジスタの指定

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	レジスタ			×	選択	周波数	
1	1	1	0				

・D_2で同期ノイズかホワイトノイズか選択します．
　D_2=0：同期ノイズ
　D_2=1：ホワイトノイズ
・D_1, D_0でノイズの周波数を設定します．

D_1	D_0	周　波　数
0	0	6.99（$=3.58\times10^3/512$）kHz
0	1	3.49（$=3.58\times10^3/1024$）kHz
1	0	1.75（$=3.58\times10^3/2048$）kHz
1	1	トーンジェネレータ＃2出力

表3・18　ノイズの設定

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	レジスタ			減　衰　量			
	R_2	R_1	R_0				

音量は音の減衰量で指定します．

D_3	D_2	D_1	D_0	減 衰 量 (dB)
0	0	0	0	0
0	0	0	1	2
0	0	1	0	4
0	0	1	1	6
0	1	0	0	8
0	1	0	1	10
0	1	1	0	12
0	1	1	1	14
1	0	0	0	16
1	0	0	1	18
1	0	1	0	20
1	0	1	1	22
1	1	0	0	24
1	1	0	1	26
1	1	1	0	28
1	1	1	1	OFF

注）レジスタのデータについては，周波数設定の項参照下さい．

表3・19　減衰量の設定

●ノイズモード0，1，1を指定すると出力周波数は，

$f=N/(32\times n\times16)$

となり低いパルス音も出力することができます．

●ノイズモードの指定でシフトレートをトーン2の出力を選択する際，トーン2の出力はオフにして下さい．

c）　減衰量の設定

減衰量は表3・19のように1バイトデータで設定します．

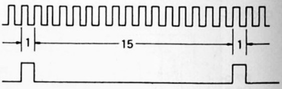


図3・46　ノイズの出力波形

(4)　PSGのソフトウェア2

それでは，実際のプログラミングについて説明していきます．まずPSG 1より440 Hzの周波数の音を出力することを考えていきましょう．この場合簡単のためにトーン0のみについて考えます．

●周波数の設定

分周比nは　$n=3.58\times10^6/(32\times440)\fallingdotseq254$

となり，これを10 bitの2進数に直すと

F_9 F_8 F_7 F_6 F_5 F_4 F_3 F_2 F_1 F_0
0 0 1 1 1 1 1 1 1 1

と表わされます．したがって書込みデータは

第1バイト　1000 1110………$8E
　トーン周波数レジスタ指定　周波数データの下位4バイト（F_3～F_0）

第2バイト　0000 1111………$0F
　↓ 0でも1でもよい　周波数データの上位4バイト（F_9～F_4）

となります．

●アッテネーションの設定

音量は最大とします．したがって減衰量は0 dBとなり書込みデータは次のようになります．

1001 0000………$90
　トーン0アッテネーションレジスタ指定　減衰量0dB

音を発生させるにはこれらを順次PSG 1のI／Oポート＄F2より出力してやればよいわけです．ただし，このままでは音が出たままになります．音を止めるには，減衰量を最大（OFF）とすればよいわけですから，アッテネーションデータとして

1001 1111………$9F
　トーン0アッテネーションレジスタ指定　減衰量MAX（OFF）

を書込みます．

これを実際に行った例を示します（リスト3・2）．この例でWAITは適度な時間待ちを意味します．

先程の例では，一定の音を鳴らすだけのものでしたが，もう少し変化をつけてみましょう．それでは，エンベロープを変えてみることにします．これは減衰量を段階的に変えてやることにより行

```
** TELESYSTEMS MAC-10D  VER 1.2  <SAMPLE1>                     PAGE 01   09.26.84

2000                  ;--- SAMPLE PROGRAM 1 ---
2000 3E8E                     LD      A,8EH
2002 D3F2                     OUT     (F2H),A           ;FIRST Byte SET
2004 3E0F                     LD      A,0FH
2006 D3F2                     OUT     (F2H),A           ;SECOND Byte SET
2008 3E90                     LD      A,90H
200A D3F2                     OUT     (F2H),A           ;OUTPUT MAX
200C                  ;
200C CD0000      E            CALL    WAIT
200F                  ;
200F 3E9F                     LD      A,9FH
2011 D3F2                     OUT     (F2H),A           ;OUTPUT MIN
2013                          END
```

リスト3・2　音を出して，止める

```
** TELESYSTEMS MAC-10D  VER 1.2  <SAMPLE2>                     PAGE 01   09.26.84

2000               ; *** SAMPLE PROGRAM 2 ***
2000               ; --- ENVELOPE CONTROL ---
2000 3E8E          START:  LD      A,8EH
2002 D3F2                  OUT     (F2H),A          ;FREQUENCY FIRST Byte
2004 3E0F                  LD      A,0FH
2006 D3F2                  OUT     (F2H),A          ;FREQUENCY SECOND Byte
2008 3E9F                  LD      A,9FH
200A D3F2                  OUT     (F2H),A          ;OUTPUT OFF
200C               ;
200C               ;ENVELOPE 1
200C 060F                  LD      B,0FH
200E 3D            ENV1:   DEC     A
200F D3F2                  OUT     (F2H),A
2011 CD3A20                CALL    DLY
2014 10F8                  DJNZ    ENV1
2016               ;ENVELOPE 2
2016 0604                  LD      B,04H
2018 3C            ENV2:   INC     A
2019 D3F2                  OUT     (F2H),A
201B CD3A20                CALL    DLY
201E 10F8                  DJNZ    ENV2
2020               ;ENVELOPE 3
2020 0610                  LD      B,10H
2022 CD3A20        ENV3:   CALL    DLY
2025 10FB                  DJNZ    ENV3
2027               ;ENVELOPE 4
2027 060B                  LD      B,0BH
2029 3C            ENV4:   INC     A
202A D3F2                  OUT     (F2H),A
202C CD3A20                CALL    DLY
202F CD3A20                CALL    DLY
2032 CD3A20                CALL    DLY
2035 10F2                  DJNZ    ENV4
2037               ;
2037 C30020                JP      START
203A               ;SOFTWARE TIMER
203A C5            DLY:    PUSH    BC
203B 0EFF                  LD      C,FFH
203D 0600          LOOP2:  LD      B,00H
203F 00            LOOP1:  NOP
2040 10FD                  DJNZ    LOOP1
2042 0D                    DEC     C
2043 20F8                  JR      NZ,LOOP2
2045 C1                    POP     BC
2046 C9                    RET
2047               ;
2047                       END
```

リスト3・3　エンベロープを使う

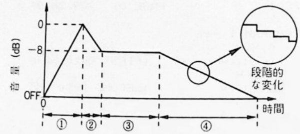
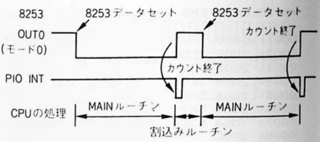


図3·47　エンベロープとは



います．

図3·47のようなエンベロープを作ってみましょう．出力する周波数は先程の440 Hzで，PSG 1トーン0だけを使用します．リスト3·3中のサブルーチン「DLY」でエンベロープの時間制御を行っており，Cレジスタに入れるデータを変えることで音の印象はかなり変わってきます．また，このエンベロープを変えることで，周波数のコントロールもできることになります．

(5)　**割込み処理**

MZ-1500では，8253とPIOにより割込み処理ができるようになっていますが，これはPSG用として作られたものです．音を出す場合，その時間的な処理は非常に大切なものとなります．前述の例ではその時間処理をソフトウェアタイマにより行いましたが，この方法では音を出している間，CPUは他の処理が全くできないことになります．そこで時間処理を8253にまかせて一定時間ごとに割込みがかかるようにしてやれば，CPUの負担は大幅に減り，その間に他の処理を行うことができるので大きなメリットがあります．

それではその割込みのシーケンスについて説明していきましょう(図3·48)．

8253カウンタ0にデータを書込むと（$E004）そのデータのカウント終了後にカウンタ0の出力は“L”から“H”に立上がります．この出力はPIOのポートAのA 4に接続されていますので，PIOがこのA 4の状態により割込みを発生できるようにしておけば，CPUに対して割込みを発生します．割込みルーチンでは，PSGにデータ書込み後8253にカウントデータを書込み，MAINルーチンへ帰してやります．(8253のカウンタ0のクロック入力は895 kHzですので，タイマとしては最大約70 mSとなります．)

リスト3·4に割込みを使ったプログラム例を示します．このプログラムはPSG 1のトーン0の周波数を最小から最大まで順番に変化させていくも

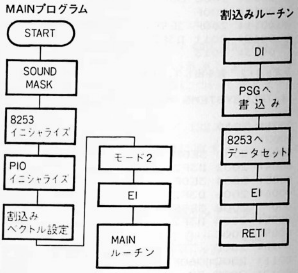


図3·48　割込みシーケンス

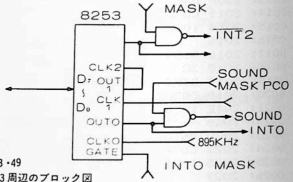


図3·49
8253周辺のブロック図

のです．割込みは1mSごとにかかってきますが，割込みの回数をカウントし，3回ごと(約3 mS)にPSGに周波数データを書込んでいきます．また，MAINルーチンはROMモニタのコマンド待ちとしていますが，コマンドによっては動作がおかしくなることがあります．これは，コマンドの中でDI(割込み禁止)あるいは8253の再設定を行っているからです．

8253（Program Interval Timer）周辺

8253は，そのカウンタ#0で音を発生させ#1#2で内蔵時計の役割をはたしています(図3·49)．

カウンタモード

#0……方形波ジェネレータ　MODE 3

#1……レートジェネレータ　MODE 2

#2……インタラプトオン・ターミナルカウント

カウンタ#0は，895 kHz の入力パルスをカウントし，指定された分周比（音符データ）で分周されて音を発生させ，サウンド IC（76489 AN）Left -ch の Audio・IN に接続され，ミキシング　オーディオ　アンプに接続されています．

このカウンタの出力は，8255 の Port CPc によりゲートされています．また，カウンタのゲートは $E 008 の D 01 によって制御されています．

カウンタ#0の出力は，割込み制御 INT 0 としても使用されていて，その出力は Z 80 PIO の Port A

```
0000                    ; *** SAMPLE PROGRAM 3 ***
0000                    ; --- PSG INTERRUPT PROGRAM ---
2000                            ORG     2000H
2000 F3                 START:  DI
2001 ED5E                       IM      2               ;MODE 2
2003 3E01                       LD      A,01H
2005 3208E0                     LD      (E008H),A       ;GATE ON
2008 AF                         XOR     A
2009 3202E0                     LD      (E002H),A       ;SOUND MASK
200C 3E04                       LD      A,04H
200E 3202E0                     LD      (E002H),A       ;INT MASK
2011 21A020                     LD      HL,VECTOR
2014 7C                         LD      A,H
2015 ED47                       LD      I,A
2017                    ;PIO INITIALIZE
2017 7D                         LD      A,L
2018 0EFC                       LD      C,FCH
201A ED79                       OUT     (C),A           ;VECTOR SET
201C 3ECF                       LD      A,CFH
201E ED79                       OUT     (C),A           ;MODE SET
2020 3E3F                       LD      A,3FH
2022 ED79                       OUT     (C),A           ;I/O SET
2024 3E37                       LD      A,37H
2026 ED79                       OUT     (C),A           ;INT CONTROL
2028 3EEF                       LD      A,EFH
202A ED79                       OUT     (C),A           ;MASK SET
202C                    ;8253 INITIALIZE
202C 3E30                       LD      A,30H
202E 3207E0                     LD      (E007H),A       ;8253 MODE SET
2031 CD7F20                     CALL    SET53
2034                    ;PSG DATA SET
2034 CD5E20                     CALL    PSGE
2037 3E90                       LD      A,90H
2039 D3F2                       OUT     (F2H),A         ;OUTPUT MAX
203B 3E83                       LD      A,83H
203D ED79                       OUT     (C),A           ;PIO INTERRUPT OK.
203F FB                         EI
2040                    ;
2040                    ;MAIN ROUTINE
2040                    MAIN:   ENT
2040 C304E8                     JP      E804H           ;JP ROM Monitor
2043                    ;
2043                    ;INTERRUPT ROUTINE
2043                    INT:    ENT
2043 F3                         DI
2044 F5                         PUSH    AF
2045 E5                         PUSH    HL
2046 CD5120                     CALL    PSG             ;PSG DATA SET
2049 CD7F20                     CALL    SET53           ;8253 DATA SET
204C E1                         POP     HL
204D F1                         POP     AF
204E FB                         EI
204F ED4D                       RETI
2051                    ;
2051                    PSG:    ENT
2051 218C20                     LD      HL,TIMES
2054 35                         DEC     (HL)
2055 C0                         RET     NZ
2056
2056 2A8A20                     LD      HL,(DATA)
```

リスト3・4　割込みを使った例

```
** TELESYSTEMS MAC-10D  VER 1.2  <SAMPLE3>                PAGE 02    09.26.84
2059 2B                    DEC    HL
205A 7D                    LD     A,L
205B B4                    OR     H
205C 2008                  JR     NZ,PSG1
205E 210004        PSGE:   LD     HL,0400H
2061 228A20                LD     (DATA),HL
2064 1813                  JR     RET1
2066 228A20        PSG1:   LD     (DATA),HL
2069 7D                    LD     A,L
206A E60F                  AND    0FH
206C C680                  ADD    A,80H
206E D3F2                  OUT    (F2H),A          ;FIRST Byte SET
2070 29                    ADD    HL,HL            ;LEFT SIFT
2071 29                    ADD    HL,HL            ;LEFT SIFT
2072 29                    ADD    HL,HL            ;LEFT SIFT
2073 29                    ADD    HL,HL            ;LEFT SIFT
2074 7C                    LD     A,H
2075 E63F                  AND    3FH
2077 D3F2                  OUT    (F2H),A          ;SECOND Byte SET
2079 3E03          RET1:   LD     A,03H
207B 328C20                LD     (TIMES),A
207E C9                    RET
207F               ;8253 SET (1mS count)
207F               SET53:  ENT
207F 3E7F                  LD     A,7FH
2081 3204E0                LD     (E004H),A        ; L Byte
2084 3E03                  LD     A,03H
2086 3204E0                LD     (E004H),A        ; H Byte
2089 C9                    RET
208A               ;
208A               DATA:   ENT
208A                       DEFS   2                ;FREQUENCY DATA (10bit)
208C               TIMES:  ENT
208C                       DEFS   1                ;INTERRUPT COUNT DATA
208D               ;
20A0                       ORG    20A0H
20A0 4320          VECTOR: DEFW   INT
20A2                       END
```

リスト3・4　割込みを使った例

のA_4に接続されています．

カウンタ#1は15.6 Hzのパルスをカウントしsec毎にOUT 1にパルスを出力します．カウンタ#2はそのパルスをカウントした後，OUT 2を"H"にします．OUT 2の出力は，割込み制御INT 1としてZ80 PIOのPortAのA_3に接続されています．またOUT 2の出力は，ゲートを通って$\overline{\text{INT 2}}$となってCPUのINTに接続されています．

(6)　PSGの応用について

音を構成する要素として，次のようなものがあります．

- ○音の高低（周波数）
- ○音色（波形，倍音構成）
- ○音の強弱
- ○音の鳴り方（エンベロープ）

それではこれらの要素とPSGとの対応を考えてみましょう．

① 音の高低

トーンおよびノイズジェネレータの周波数設定によりコントロール

② 音色

PSG出力はデューティ50％のパルスのみですが，3周波数のミキシングおよびアッテネータによりコントロール

③ 音の強弱

各ジェネレータのアッテネーションによりコントロール

④ 音の鳴り方

アッテネーションを，時間的に変化させてやることによりコントロール．

以上のようにPSGはすべての要素を兼ね備えていることがわかると思います．したがって無限に近い種類の音を出力させることができますが，音は実際に聞かないとわからないため，どうしても試行錯誤が必要となります．

第4章

MZ-1500の応用

4-1：PCGの使いこなし

PCGの重ねあわせ
使用上の注意

4-2：パターンエースの使いかた

エディタの起動
大きな絵を描く

4-3：ミュージックエースの活用

音符データの入力方法
サウンド作りに挑戦
実際の作成例

4-1
PCGの使いこなし

4-1-1
PCGの重ねあわせ

リスト 4・1 が PCG を重ね合わせるプログラムです．まず簡単な説明からしましょう．

CALADR：画面上のアドレスを求める．
PEKADR：　　〃
POPDAT：PCGDATA を読出す．
MIXDAT：重ね合わせ処理をします．
PUSDAT：PCGDATA を書込む．

くわしい説明は後にありますのでここではこれぐらいにしておきます．それでは，このプログラムを使用した例をあげてみましょう．リスト 4・2 がその例になります．この例は BASIC であらかじめ定義してある PCG 1，2，3，4 と 5，6，7，8 を重ね合わせて PCG 9，10，11，12 に定義するプログラムです．機械語を使用していますので，DATA 文の打ち込みには注意して下さい．特に，RUN させる前には，一度 QD にセーブしておいた方がいいでしょう．

RUN させてみて下さい．PCG が合成されたものが表示されたはずです．もし何もおこらなかったり，暴走するようなら，もう一度リストを確認した方がいいでしょう．

それでは，プログラムの説明に移りましょう．

```
CF80                          ;
CF80                          ;
CF80                          ;
CF80                          ;
CF80                          ;
CF80                          ;
CF80                          ;
CF80                          ;
CF80                          ;
CF80                          ;
CF80 2100D4           LD     HL,D400H       ; PCG ADDRESS SET
CF83 CD02CF           CALL   CF02H          ; CALADR
CF86 3E00             LD     A,00H          ; SKIP DATA = 0
CF88 011002           LD     BC,0210H       ; XY SET
CF8B 1180CE           LD     DE,CE80H       ; BACK DATA LOAD ADDRESS
CF8E CD04CF           CALL   CF04H          ; POPDAT
CF91 D9               EXX                   ; CHANGE
CF92 2104D4           LD     HL,D404        ; PCG ADDRESS SET
CF95 CD02CF           CALL   CF02H          ; CALADR
CF98 1100CE           LD     DE,CE00H       ; FRONT DATA LOAD ADDRES
S
CF9B 011002           LD     BC,0210H       ; XY SET
CF9E EEAF             XOR    AF             ; SKIP DATA = 0
CFA0 CD04CF           CALL   CF04H          ; POPDAT
CFA3 D9               EXX                   ; CHANGE
CFA4 011002           LD     BC,0210H       ; XY SET
CFA7 11D0CF           LD     DE,CFD0H       ; MASK DATA ADDRESS
CFAA CD06CF           CALL   CF06H          ; MIXDAT
CFAD EB               EX     DE,HL          ; CHANGE DE,HL
CFAE 2108D4           LD     HL,D408H       ; PCG ADDRESS SET
CFB1 CD02CF           CALL   CF02H          ; CALADR
CFB4 011002           LD     BC,0210H       ; XY SET
CFB7 EB               EX     DE,HL          ; CHANGE DE,HL
CFB8 AF               XOR    A              ; SKIP DATA = 0
CFB9 CD08CF           CALL   CF08H          ; PUSDAT
CFBC C9               RET
CFBD                          ;
CFBD                  END
```

リスト 4・1　機械語サブルーチン

まず BASIC プログラムの方からはじめます．

行番号 200 からはマスク DATA をセットするプログラムです．READ 文に対応する DATA は 2080 行です．

行番号 300 からは機械語プログラムを書込むプログラムです．DATA は文字データとして扱っているので，VAL 文で 16 進数に変換しながらメモリに書込んでいます．

このDATA が今回のプログラムのメインである PCG の定義や重ね合わせをします．

行番号 400 からは機械語のサブルーチンをサポートするためのプログラムですが，このプログラムも機械語で書かれています．

行番号 100 からは PCG を定義するためのプログ

```
10 REM ----------------------------
20 REM ---                      ---
30 REM --- PCG FONT MIX SAMPLE ---
40 REM ---                      ---
50 REM ---          B.Y          ---
60 REM ---                      ---
70 REM ---    DIGITAL ｶﾏｷﾘ      ---
80 REM ---                      ---
90 REM ----------------------------
100 REM --------------------
110 REM --- PCG FONT SET ---
120 REM --------------------
130 INIT "CRT:I":LIMIT $CE00
140 FOR A=1 TO 8
150 READ A$
160 FONT$(2,A)=HEXCHR$(A$)
170 NEXT
180 FPRINT [0,0] 1,2,0,0,5,6,0,0,9,10
190 FPRINT [0,1] 3,4,0,0,7,8,0,0,11,12
200 REM ---------------------
210 REM --- MASK DATA LOAD ---
220 REM ---------------------
230 READ A$
240 FOR A=1 TO 32
250 POKE $CFCF+A,VAL("$"+MID$(A$,A*2-1,2))
260 NEXT
300 REM --------------------- .
310 REM --- ｷｶｲｺﾞ LOAD 1 ---
320 REM ---------------------
330 FOR A=1 TO 128
340 READ A$
350 POKE $CEFF+A,VAL("$"+A$)
360 NEXT
400 REM ---------------------
410 REM --- ｷｶｲｺﾞ LOAD 2 ---
420 REM ---------------------
430 FOR A=1 TO 60
440 READ A$
450 POKE $CF7F+A,VAL("$"+A$)
460 NEXT
500 REM ---------------
510 REM --- PCG MIX ---
520 REM ---------------
530 USR($CF80)
540 END
1000 REM ----------------------
1010 REM --- MAKE PCG DATA ---
1020 REM ----------------------
1030 PRINT "■!":FOR A=1 TO 8
1040 A$=FONT$(2,A)
1050 PRINT 1990+A*10;" DATA ";
1060 FOR B=1 TO 24
1070 PRINT RIGHT$("0"+HEX$(ASC(MID$(A$,B,1))),2);
1080 NEXT:PRINT
1090 NEXT
```

リスト 4・2　PCG フォントミックスのサンプルプログラム

ラムで，PCG 1～8を定義します．定義DATAは行番号2000～2070行です．

結果的にいえば，BASICプログラムはDATAの定義とプログラムの書込みしか行なっていませんのであまり参考にならないかもしれませんが，メインであるPCGの重ね合わせができる機械語サブルーチンは役にたつはずです．

それでは次に機械語サブルーチンの使用条件を説明します．

機械語サブルーチンについて

<CALADR>

HLにセットされているPCGNO.から（0～03FFH）PCGRAM上のアドレスを求めます．例としてはHLに0をセットしてCALL CALADRを実行するとHLにはD000Hが帰ってきます．1ならばD008H，100HならばD800Hという具合になります．つまり，定義したり変更したりする時のアドレス計算をしてくれるサブルーチンです．

<PEKADR>

結果的にはPCGRAMのアドレスを求めるプログラムですが，CALADRがHLにPCGNO.をセットしていたのに対して，今度は，PCG画面（D400H～D7FFH）のアドレスをHLにセットします．つまり，HLが示すPCGのPCG定義アドレスを

```
1100 REM -----------------------
1110 REM --- MAKE MASK DATA ---
1120 REM -----------------------
1130 PRINT 2080;" DATA ";
1140 FOR A=13 TO 16
1150 A$=FONT$(2,A)
1160 FOR B=17 TO 24
1170 PRINT RIGHT$("0"+HEX$(ASC(MID$(A$,B,1))),2);
1180 NEXT
1190 NEXT:PRINT "■"
2000 DATA 000000000000003F00000F1F3F7FFF3F000000000000003F
2010 DATA 00000000000000FC0000F0F8FCFEFFFC00000000000000FC
2020 DATA 3F3F3F3F7FBF55AA3C3C3C3CFFFFFFFF3F3F3F3FFFFFFFFF
2030 DATA FCFCFCFCFDFE55AA0C0C0C0CFFFFFFFFFCFCFCFCFFFFFFFF
2040 DATA 0000000000181818000000000018181800 3C7E7E7E666666
2050 DATA 0000000000000040000000000040E0400000000000040E04000
2060 DATA 18181818181818181818181818181818240000000000000000
2070 DATA 40400000000000001C1C483E0808142200000000000000000
2080 DATA FFC3818181818181FFFFFFFFBF1FBFBFC3E7E7E7E7E7E7E7A3A3B7C1F7F7EBDD
3000 REM -----------------------
3010 REM --- ｷｶｲｺﾞ ｻﾌﾞﾙｰﾁﾝ 1 ---
3020 REM -----------------------
3030 DATA 18,18,18,06,18,1C,18,39
3040 DATA 18,57,D3,E3,4E,7C,C6,08
3050 DATA 67,7E,07,07,E6,03,67,69
3060 DATA D3,E1,29,29,29,3E,D0,84
3070 DATA 67,C9,D5,32,32,CF,3E,03
3080 DATA D3,E5,C5,E5,C5,06,00,ED
3090 DATA B0,0E,00,09,C1,10,F5,E1
3100 DATA C1,3D,C2,28,CF,E1,D3,E6
3110 DATA C9,E5,3E,03,C5,D5,08,C5
3120 DATA 1A,A6,D9,B6,23,D9,77,13
3130 DATA 23,0D,C2,48,CF,C1,10,EF
3140 DATA D1,C1,08,3D,C2,44,CF,E1
3150 DATA C9,32,70,CF,3E,03,D3,E5
3160 DATA C5,D5,C5,06,00,ED,B0,0E
3170 DATA 00,EB,09,EB,C1,10,F3,D1
3180 DATA C1,3D,C2,66,CF,D3,E6,C9
3200 REM -----------------------
3210 REM --- ｷｶｲｺﾞ ｻﾌﾞﾙｰﾁﾝ 2 ---
3220 REM -----------------------
3230 DATA 21,00,D4,CD,02,CF,3E,00
3240 DATA 01,10,02,11,80,CE,CD,04
3250 DATA CF,D9,21,04,D4,CD,02,CF
3260 DATA 11,00,CE,01,10,02,AF,CD
3270 DATA 04,CF,D9,01,10,02,11,D0
3280 DATA CF,CD,06,CF,EB,21,08,D4
3290 DATA CD,02,CF,01,10,02,EB,AF
3300 DATA CD,08,CF,C9
```

リスト4・2　PCGフォントミックスのサンプルプログラム

求めるサブルーチンです.

<POPDAT>

このプログラムは, PCGRAM より PCG の定義データを, メイン RAM 上に読出してくるプログラムです. 読み出してくる順番は, 緑, 赤, 青の順になります. サイズは, B レジスタが y キャラクタの数, C レジスタが x キャラクタの数に 8 をかけたもので B×C バイト×3面分(緑, 赤, 青)となります. このデータは DE レジスタ以降に読出されます.

<MIXDAT>

2つの PCG が重なる時に使用します.

まず, HL には重なった時に後ろにくる DATA を, HL′には重なった時, 前にくる DATA を, DE には重なる所を削除するためのマスクデータをセットします. この時のPCG DATAはあらかじめ "POP DATA" を使用してメインメモリ上に読出した物を使用します. もちろんあらかじめ用意してあった物を使用してもかまいません. 結果は HL 以降に書込まれます. つまり, 後ろに来る DATA は, こわれることになりますので注意して下さい.

<PUSDAT>

このプログラムは"POPDATA"や"MIXDAT"を使用してできあがった PCGDATA を DE レジスタで示す PCG 定義アドレス以降に書込む(定義する)プログラムです.

次に具体的にサブルーチンの仕様について触れてみます.

① CALADR　HL にセットされた PCGNO. 0～03FFH から PCGRAM 上のアドレスを求める.

IN：HL　PCGNO.
OUT：HL　PCG アドレス
BREAK：AF

② PEKADR　HL が指し示す PCG の定義データが入っているアドレスを求める.

IN：HL　(HL)－
OUT：HL　PCG アドレス
BREAK：AF, C

③ POPDAT　HL のアドレスから C×B バイト分の PCGRGBDATA を 3, 2, 1 バンクの順に DE 以降に読出す.

IN：A　行から行へ変る時のスキップバイト数
B　y キャラクタの数
C　x キャラクタの数×8
DE　転送先　3, 2, 1面の順
HL　読出しアドレス(CALADR・PEKADR で求めたアドレス)
OUT：HL ←転送先アドレス (元 DE の値)
BREAK：AF, DE (A＝0)

④ MIXDAT　HL が指し示すアドレス以降の DATA と DE 以降のマスク DATA との AND をとり, HL′以降の DATA と OR をとり (HL) に書込む.

IN：B　y キャラクタの数
C　x キャラクタの数×8
DE　マスク DATA
HL　PCGDATA 1 (後ろになる) → ANSER
HL′　PCGDATA 2 (前になる)
OUT：HL 合成 DATA (PCGDATA と同じ所に書かれる)
BREAK：AF, AF′ (A＝0), BC

⑤ PUSDAT　HL 以降の DATA　B×C　バイト分をDEの示すPCG定義アドレスへ 3, 2, 1 画面画面の順に転送する.

IN：A　次の PCG 列まで DE をスキップさせる.
B　y キャラクタの数
C　x キャラクタの数×8
DE　PCG 定義開始アドレス
HL　PCG 定義 DATA
OUT：DE 以降に PCG を定義する
BREAK：AF, HL (A＝0)

4-1-2 使用上の注意

サブルーチンを使用する時に注意する事があります. まず X 方向の最大値は 31 キャラクタ(31×8＝248 バイト) までです.

Y 方向は 256 キャラクタですが, まずオーバーする事はないと思います. PCG メモリから読出してくるデータは B レジスタ×C レジスタ×3 バイトありますので気をつけて下さい.

POPDAT と PUSDAT に出てくる SKIP バイト数 (A レジスタの値) については, 例を上げてみますと 2×2 キャラクタの PCGDATA を MZ

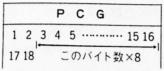


図4・1
データの読出し

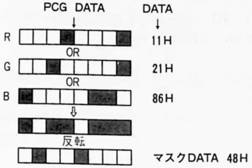


図4・2 マスクデータの考えかた

-1500 本体付属の PCG エディタ(パターンエース)で作ったとしましょう．普通に書込めば PCG 1，2，17，18 となります．(もっともつづけて書込む方法もありますが……，)そのデータを読出す時にAレジスタの値を 14×8バイトにセットしておけばメモリ上にはつながった DATA が読出されます(図4・1)．参考までに 3×3 の大きさだと Aレジスタの値は 13×8バイトになります．

マスクデータについて

次にマスクデータの話に移ります．ここでいうマスクデータとは，サブルーチン MIXDAT で使用されるデータの一つで，2つの PCGDATA が本当に重なってしまうのを防ぐための重要なデータです．ここで一つの実験をして下さい．行番号 2080 行のマスクデータを全部 FF に書き変えて下さい．そしてもう一度 RUN してみて下さい．どうですか？ずいぶんおかしな状態になったはずです．マスクデータとは，重なった時にバックにくる不必要なデータを削除するための物です．では，そのマスクデータの具体的な作り方を説明すると，重なった時に前の方にこさせたり PCG の RGB データのすべての OR をとり(図4・2参照)そのデータをすべて反転する事により出来あがります．つまり，画面に表示した時に，黒以外の所をすべてカットする事になりますが，ここで気を付けてほしいのが，その PCG の中で黒色を使用している時で，その所のバックデータがのこってしまう事になりますので，必要に応じてマスクデータを変更して下さい．またこの事を利用して，たとえば「車の窓のむこう側がみえる」といった事もできますので，いろいろ工夫してみて下さい．

それでは，PCG エディタを使って実際にマスク DATA を作ってみます．

① PCG エディタを起動する
ここでロードする PCG ファイルネームを"ナシ"にすると現在定義されている PCG がそのまま残る．

② 前と後ろの PCGDATA を 2×2 の大きさで作り，後ろの DATA を 1～4 に書込み前のデータを 5～8 に書く．

③ 前に来る DATA を R コマンドを使って黒以外の色を白にする．ここで黒色を使っていたら，その部分も白にしておく．

④ R コマンドで黒を緑に，白を黒に変える．

⑤ 〔F 7〕で PCG 13～16 に定義する．

⑥ プログラムを BREAK で止めて，リスト1をロードする．

⑦ RUN 1000 を実行する．

⑧ READY.のメッセージが出たら，〔CR〕キーを 9 回押して RUN させる

⑨ 画面に合成された PCGDATA が表示される．

以上がマスク DATA 作成についての説明です．

1～5がマスク DATA の作成で，以降はそれを使用した重ね合わせのための DATA 部の書込みを行なっています．最後になってしまいましたが，機械語サブルーチン 2 の説明をします．

リスト 4・2 をもう一度見て下さい．

11，12 行で D 400 H (PCG 画面の左上すみ)にある PCGDATA アドレスを HL に求めます．

18～19，29～30 行についても同じです．

13，22，33 行では前に説明した SKIP バイト数のセットを行なっています．今回はデータがつながっているので 0 にセットされます．

14，21，25，31 行は X と Y のサイズのセットです．2×2 の大きさですので，B レジスタには 2 を，C レジスタには 2×8 の 16 バイト (10 H) がセットされます．

15，20 行は，サブルーチン POPDAT から読出されるデータの転送先です．

17 行では，レジスタの切換えをしています．

24 行はその逆で HL レジスタをもとにもどします．この時に HL′には，前に来る PCGDATA の先頭アドレスが入ります．26 行は，マスクデータの先頭アドレスをセットしています．

27 行は PCG の重ね合せの処理をします．結果は HL レジスタ以降にセットされています．

28～34 行で HL レジスタ以降のデータを PCG に定義しなおします．

4-2
パターンエースの使いかた

4-2-1
エディタの起動

あなたは，コンピュータで使っている絵や音楽などを自分の好きなように変えたり，新しくつくってみたいと思いませんか？このパターンエース・ミュージックエースは，そんな夢をかなえてくれるソフトウェアなのです．あなたのイメージも思いのままに作りだせるのです．

それでは,さっそくパターンエース（PCG EDIT）から動かしてみましょう．

パターンエースでは，PCG EDIT，PCG 1.00, PCGM 1.00, PCGE 1.00 の 4 つのプログラムを使用します．

○ PCG エディタの起動

① BASIC を起動します．

② NEW ON 2 CR と打ち込みます．

この命令で,いま必要のない BASIC の機能を削り，その分だけユーザープログラムエリアを大きくします．

③“PCGEDIT”，“PCGE 1.00”，“PCGM 1.00”と“PCG 1.00”が全部入っている QD をセットします．（わからないときは，DIR 命令で探しましょう．）

④ RUN “PCGEDIT” CR と打ち込みます．

しばらく QD が動いたあとで，

PCG エディタ VERI.0A
データの QD をいれてキーをおしてください

というメッセージが表示されます．

パターンエースでつくるデータは,QD に新しいファイルとして記録します．そこで，データファイル用に QD を用意してください．はじめてデータをつくる場合は，初期化したメディアを入れてください．(初期化の方法は BASIC LANGUAGE MANUAL の 70 ページまたは 137 ページを参照)以前に作ったデータの変更，追加を行う場合は，そのデータのある QD をセットしてからキーを押してください．すると，セットした QD のディレクトリ（QD に入っているファイルの名前一覧）が表示され，次のように入出力ファイル名を聞いてきます．

PCG ニュウリョク：ナシ

はじめてデータをつくる場合には，入力ファイルがないので，そのまま CR を押します．以前に作ったデータを使う場合は，ディレクトリを参考に，そのファイル名を入力します．すると，

マップニュウリョク：ナシ

と表示されます．PCG ニュウリョクと同じようにします．（＊マップとは，PCG の各キャラクタを使って大きな画面の絵を作ったりするときの指定用の画面データです．）次は，

PCG シュツリョク：FONT

と表示されます．これは，今から作るデータを記録するファイルの名前を指定するものです．他のファイルとまちがわない名前にしておきましょう．ここでは，“MYFONT”というファイル名にしておきましょう．また，ここで実際につくられるファイルは，BRG 3 色分になり，MYFONT 1，MYFONT 2，MYFONT 3 の 3 つです．ニュウリョクのときは，MYFONT だけで，この 3 つのファイルを読込みます．このようにファイルが 3 つに分かれるのは，この PCG データだけです．

マップシュツリョク：MAP

ここでも，やり方は同じです．“MYMAP”と入力しておきましょう．

マップ入力をナシにした場合，新しいマップの大きさを決めなければなりません．大きさはタテ，ヨコともに 24 キャラクタ以上で，タテ×ヨコが 6,000 を越えないようにします．タテ，ヨコの順に打ち込みます．

マップ　タテ　25
マップ　ヨコ　40

以上の入力が終わると，確認のためのメッセージが表示されます．

これでよいですか（Y，N）？

よければ Y キーを押して次のステップに進みます．もし，まちがっていたら N キーを押して，はじめからやり直します．このとき，この 2 つ以外のキーを押すと Y キーを押したことと同じになり

PCGエディタ VER1.0A

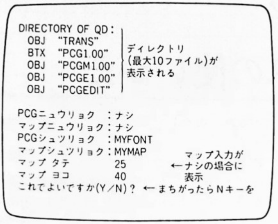


図4・3 メッセージ画面

ます（図4・3）.

入力ファイルを指定した場合，そこファイルを読み込みます．その後，

プログラムのQDをいれてキーをおしてください

と表示されることがあります．このときは，はじめに使ったQDをセットしてキーを押します．

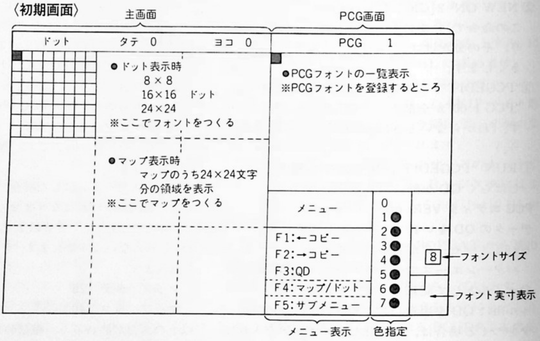
○**フォントを作る**

フォントを作る時は，ドッドモードにしなければいけません.〔F4〕キーを押して主画面をドットモードにして下さい．

画面右下のカラーの右横に数字がでています．これがフォントのサイズです(図4・4参照)．[I]キーを押すと8（8×8），16（16×16），24（24×24）を順に表示していき，フレームの大きさも変化します．ここでは16に合わせてみましょう．

右下に表示されたカラーと左横のカラーコードを見ながら数字を入れると，カーソル（点滅するチェック模様）の位置に色のついたマルが入ります.カーソルを動かすには,カーソルキー([←][→][↑][↓])

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	7	7	7	0	0	0	0	0	0	0	0	7	7	7	0
7	0	0	0	7	0	0	0	0	0	0	7	0	0	0	7
0	3	3	0	0	7	0	0	0	0	7	0	0	3	3	0
3	3	3	3	0	0	7	0	0	7	0	0	3	3	3	3
3	3	3	3	3	0	0	7	7	0	0	3	3	3	3	3
3	3	3	3	3	3	0	2	2	0	3	3	3	3	3	3
0	3	3	3	3	3	3	6	6	3	3	3	3	3	3	0
0	0	3	3	3	3	3	6	6	3	3	3	3	3	0	0
0	3	3	3	3	3	3	6	6	3	3	3	3	3	3	0
3	3	3	3	3	3	0	6	6	0	3	3	3	3	3	3
3	3	3	3	3	0	0	6	6	0	0	3	3	3	3	3
0	3	3	3	0	0	0	0	0	0	0	0	3	3	3	0
0	0	3	0	0	0	0	0	0	0	0	0	0	3	0	0
0	0	3	0	0	0	0	0	0	0	0	0	0	3	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



PCGエディタの機能

〈ドットモード〉

フォント作成 ——→ PCG一覧表へコピー

〈マップモード〉

PCGのフォントを選ぶ ——→ マップ上にコピー

図4・4 フォントサイズ

```
100 REM ******** MOVE FONT ********
110 REM
120 INIT "CRT:I"
130 CLS 3
140 PCGNUM=465
150 FOR X=0 TO 36
160 FPRINT[X,0],PCGNUM,PCGNUM+1
170 FPRINT[X,1],PCGNUM+16,PCGNUM+17
180 WAIT 90 REM0.09ﾋﾞｮｳ ﾏﾂ
190 NEXT X
200 REM
210 END
```

リスト4・3
フォントを動かす

を使います．(SHIFT)キーを押しながらカーソルキーを押すと8マス分動きます.)ためしに，左のようなフォントを作ってみましょう．

さて，どんなフォントができましたか？主画面では，大きく表示されますが，実際の大きさは，画面右下スミに表示されている大きさです．主画面で，フォントを変えると右下も変わります．

できあがったフォント．でも，このままでは使えません．使うにはPCGに登録しなければなりません．まず，〔F 2〕キーを押します．すると，PCG画面でカーソルがでます．カーソルを動かすと，PCG××という表示の数字が変わります．この数字がカーソルのある場所のPCG番号です．パターンエースは，1から1000までが使用できます．

そして，適当な場所にカーソルを動かし，もういちど〔F 2〕キーを押すと，PCGにコピーされます．ここではPCG 465にコピーしておきましょう．ではここでひと息……

今つくったフォントをBASICで動かしてみよう．この「ちょう」を，画面の左から右へ動かすプログラムがリスト4・3です．

SHIFT＋BREAKとSHIFT＋CLRを操作しリスト4・3を入力し，実行してみましょう．

「ちょう」が動きましたか？

パターンエースにもどるには，RUN 1000 CR と操作し，はじめから操作し直して下さい．フォントは覚えておりますので入力し直す必要はありません．

4-2-2

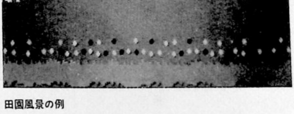
大きな絵を描く

マップとは，大きな画面のどこにどのフォントを置くか，ということを書込む図面です．限られた数のフォントでも，うまく配置すれば，美しい画面でも複雑な画面でもつくることができます．例えば，画面の背景に青空をかきたい場合には，8×8のすべてのドットが青色のフォントをひとつ作り，空の位置に置いていけば（コピー）よいのです．

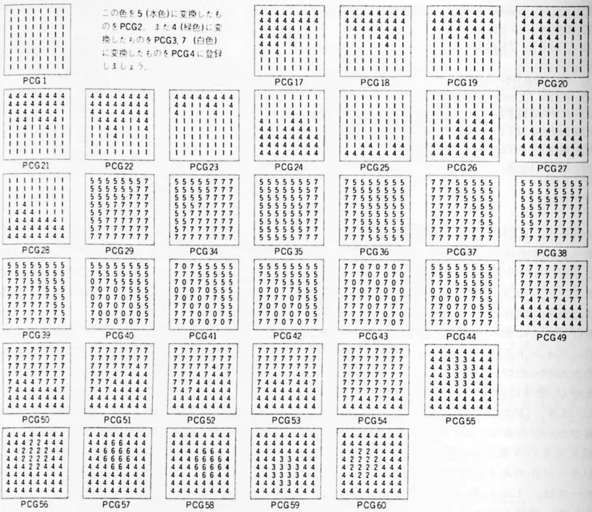
山をかくことを考えましょう．山をまるごと描こうとしたら，PCGメモリを多くとり，おまけにすべて同じ形の山になってしまって，単調です．もっとうまい方法があるので紹介しましょう．それは，山の斜面をいくつかと，山の頂上をつくり，それらをつなげていくのです．どうですか，メモリも少なく，複雑な絵もカンタンにかけるでしょう．このようにして他の要素もつくりましょう（図4・5参照）．

PCG一覧表にコピーしたフォントを，こんどはマップにコピーしていきます．

まず，主画面をマップモード（〔F 4〕キー）にしてください．そして，マップ（主画面）のカーソルを動かして，コピーする位置を決めます．それから，〔F 1〕キーを押します．するとマップ上のカーソルが変わり（＋），PCGにカーソルが移動します．そのカーソルを動かし，こんどはコピーしたいフ



田園風景の例



田園の風景の例に使用する PCG の例

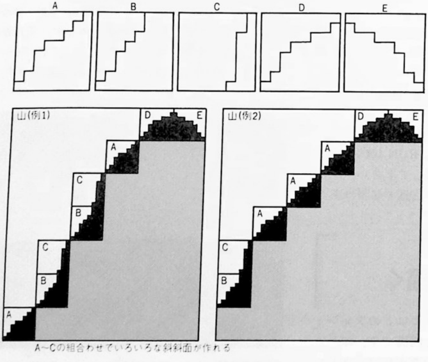


図4・5
山の斜面の作りかた

〈岸辺の花を連続させる〉コピー完了

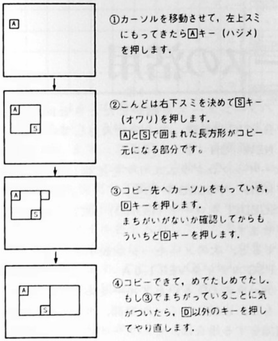


図4・6　コピーの方法

ォントの位置にもってきます．そして，もういちど〔F 4〕キーを押すと，マップの指定位置にコピーできます．ただし，コピーするフォントの大きさを，はじめにTキーで変更しておく必要があります．

しかし，1つずつコピーしていくのは不合理なのでそれをさけるにはマップ内のある部分を，同じくマップ内の他の場所へまるごとコピーする方法があります．まず，はじめに，コピーしたい部分を長方形として決めます．左上と右下のスミをそれぞれAキーとSキーで指定します．もちろんカーソルを動かしてから，それぞれのキーを押すのです．次に，コピー先へカーソルを動かします．そしてDキーを押します．するとサブメニューのDの行は反転します．ここでもういちどコピーの位置を確かめてからDキーを押すと，コピーができます（図 4・6 参照）．

この他にも，同じもののくり返しなどは，もっとうまくカンタンにできるマップフルという方法があります．

これは，マップ内の指定した領域を，すべてキャラクタでぬりつぶすものです．はじめに指定先をAとSで決めます．そして，マップ内にあるマップフルしたいキャラクター上にカーソルを移動します．それから，Fキーを押し，いちど確認してから，もういちどFキーを押します．

○ファイルする

つくったPCGとマップをQDにセーブします．それぞれをデータとしてファイルしておかなければ利用できません．〔F 3〕キーで，QDメニューを表示させてください．そして，〔F 1〕キーを押すとPCGがQDにセーブされます．さらに〔F 2〕キーを押すとマップがQDにセーブされます．ただし，ファイル名は，はじめにPCGシュツリョクとマップシュツリョクで指定した名前（MYFONT，MYMAP）になります．また，ファイル名を，ナシとしていた場合には，セーブされません．

図 4・7 にパターンエースのコマンド一覧を示しますので，これに従って操作して下さい．

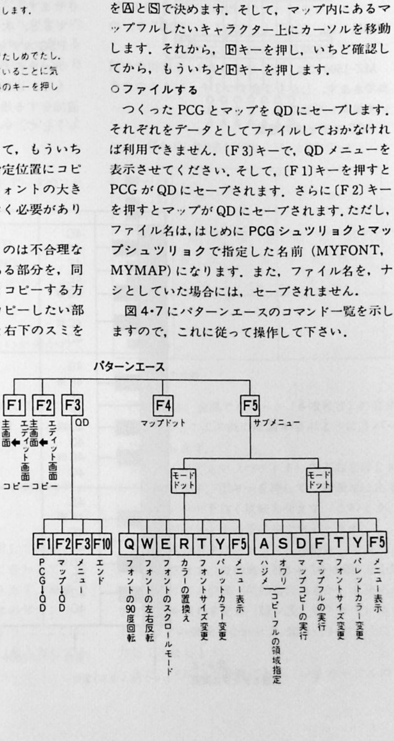


図4・7
コマンド一覧

4-3 ミュージックエースの活用

4-3-1 音符データの入力方法

次にミュージックエース（PSGEDIT）を動かしましょう。

ミュージックエースではPSGEDIT，PSG 1.00，PSGM 1.00，FONT 1，FONT 2，FONT 3，のプログラムを使用します。

MZ-1500には，PSGが2つあり，6重和音が演奏できます．しかも3音ずつ2チャンネルのステレオ演奏も可能です．

○ PCGエディタの起動

パターンエースを動かしたときと同じように BACIC起動後，次の順で入力していきます．

NEW ON 2 [CR]

ユーザープログラムエリアを拡張します．

RUN "PSGEDIT"[CR]

PSGEDITをロード（LOAD）して，その後RUNさせます．

すると，次のメッセージが表示されます．

PSG エディタ VER 1.0 A

データの QD をいれてキーをおしてください

もし，以前につくった楽譜，サウンドの変更や追加をする場合は，そのデータの入ったQDをセットして，いずれかのキーを押します．また，新

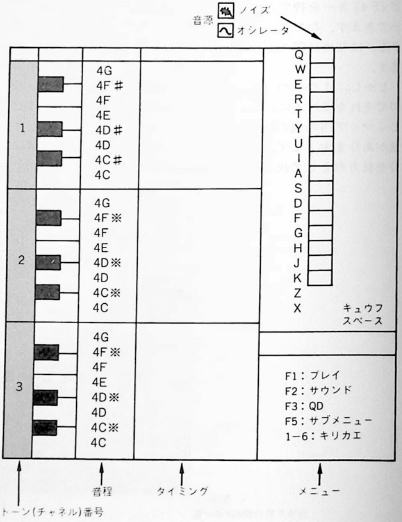


図4・8
楽譜エディタの画面

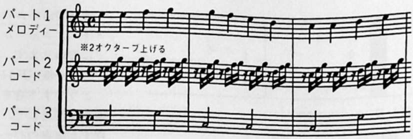


図4・9
「よろこびの歌」の例

しくデータをつくる場合には，そのままいずれかのキーを押します．ここでは，新しく楽譜データをつくるため，そのままキーを押してください．データの追加・変更する場合は，セットしたQDのディレクトリが表示されますので，まちがいのないよう入力ファイル名を入力してください．新しくデータをつくる場合は，入力ファイルがないので🄽🄾と入力します．ここでは，QDに入っているサウンドデータを利用しますので次のように入力しましょう．

ガクフニュウリョク：ナシ
　　新しく楽符データをつくるのでこのように入力する

サウンドニュウリョク：SOUND
　　QDに入っているサウンドデータのファイル名

続いて出力ファイル名を指定します．

ガクフシュツリョク：MYGAKUFU

サウンドシュツリョク：MYSOUND

以上の入力が終わると，確認のため次のメッセージが表示されます．

これでよいですか（Y，N）？

画面を見て，入力がまちがっていないか確かめてからキーを押しましょう．

なお

プログラムのQDをいれてキーをおしてくださいと表示されることがあります．このときは，はじめに使ったQDを入れてキーを押します．

〈初期画面〉

画面を見ながら，楽譜を入力していく楽譜エディタの画面です（図4・8参照）．

楽譜をMZに教える

デモエースで使うサウンドを利用して楽譜データをつくってみよう．音程・音符に合わせて，楽譜データをひとつずつ入力していきます．最高で6音まで同時に使えますので，全体のサウンドを考えて音の配分を決めておきましょう．ここでは，ベートーベン作曲「よろこびのうた」を入力してみましょう．

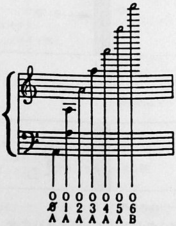


図4・10　音の高さ（例は"ラ"）と指定データ

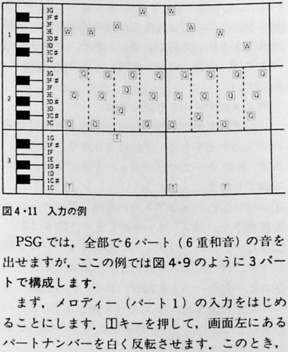


図4・11　入力の例

PSGでは，全部で6パート（6重和音）の音を出せますが，ここの例では図4・9のように3パートで構成します．

まず，メロディー（パート1）の入力をはじめることにします．🄸キーを押して，画面左にあるパートナンバーを白く反転させます．このとき，カーソルがパート1の帯の中で表示します．

音符の入力方法は，カーソル移動キーでカーソルを移動させて，音の高さ（図4・10参照）と小節の中でのリズム位置（何拍目にあるか）を決め，サウンドを表わす記号（🅀～🅇）を入れていきます．先ほどの楽譜に合わせて，図4・11のように入力してみましょう．

また，上には小節数があり，いまカーソルのあ

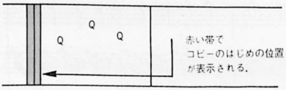


図4・12 カーソルの操作（1）

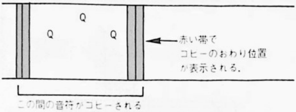


図4・13 カーソルの操作（2）

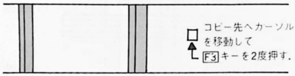


図4・14 カーソルの操作（3）

る小節は何小節目かすぐわかるようになっています．入力は最大48小節までできます．

また，1小節の中に音符は16コまでしか入りませんが，この小節は入力のための目安なので，16分音符より短い音符が必要なときや，3拍子の曲のときは，画面上の線を無視して入力すればよいのです．

同じメロディのくり返しには

メロディを他の小節へコピーすることもできるので入力がとてもカンタンになります．方法は，まず〔F5〕キーでサブメニューを表示させ，次の手順でします．

①コピーしたいメロディーの先頭位置にカーソルを移動させ，〔F1〕キーを押します（図4・12）．
②コピーしたいメロディーの最後にカーソルを移動させ，〔F2〕キーを押します（図4・13）．
③コピー先のパートを選び，コピー先の先頭へカーソルを移動させ，〔F3〕キーを押します。コピーを実行してよいかどうか確認して，もういちど〔F3〕キーを押します（図4・14）．

もしも入力をミスした場合には，入力した音符が高抵に変化しているとウインドウの画面では全部見えない場合がありますので，再度入力する場合には，入力以前に入力部分をクリアしておくと便利です．

方法は，コピーと同じようにその領域のはじめと終りを決め，〔F4〕キーを2度押せば完了です．

○プレイミュージック

それではいよいよ実際に演奏させてみましょう．メインメニューから〔F1〕キーを押すとプレイモードに変わります．

もう一度〔F1〕キーを押すと演奏開始．

どうです？音をまちがっていなかったですか．演奏に関連して，フレイモード時に[1]～[6]キーを押すと演奏のパート分け，〔F2〕キーを押すと，演奏のテンポ〔F3〕と〔F4〕キーで領域を指定し，指定領域の演奏などの機能もあります．

○ファイルする

せっかく作ったデータも，電源OFFになれば水のアワですね．つくった楽譜データやサウンドデータのセーブを忘れないようにしましょう．メインメニューから〔F3〕キーを押して，QDモードに移ります．そして，〔F1〕キーを押せば楽譜データのセーブ，〔F2〕キーを押せばサウンドデータのセーブが始まります．ただし，このときにセーブされるファイル名は，はじめに入力した出力ファイル名（楽譜データ“MYGAKUFU”サウンドデーター“SOUND”）です．もちろん，はじめに田図と入力していたら，セーブされません．また，QDに同じ名前のファイルがあるときもセーブ不能です．注意しましょう．

4-3-2 サウンド作りに挑戦

PCGには自分でサウンドをつくり出せる機能があります．

オーケストラでたとえるならバイオリンやフルート，オーボエ，コントラバス，それに打楽器といろいろあればそれだけすばらしいサウンドが作れるものです．ゲームなどの効果音も同じです．それではサウンドづくりを始めてみましょう．

〔F2〕キーを押して，サウンドエディタ画面にしてください（図4・15参照）．

画面の最上段はエンベローブ．その次に周波数ウインドウ，そしていちばん下にコマンドメニューとサウンドメニューが表示されています．

音源には，オシレータ（きれいな音）とノイズ（雑音）の2種類が選択でき，〔F6〕キーを押すことによって変更できます．白く反転している方が，現在の音源です．

音の種類は

[Q]から[K]まで画面に表示されている16のキーに

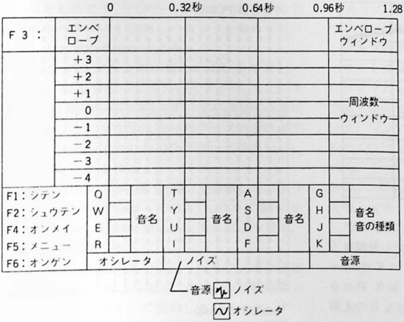


図4・15
サウンドエディタ画面

それぞれ音を定義することができます．表示は，まずキー名，そして音源が記号で表示され，その次に音名を表示するスペースがあります．そのスペースが白く反転しているのが，現在指定の対象となっているキーです．音名の入力は，まず対象となるキー（Q～K）を決めて押し，それから〔F 4〕キーを押して音名入力モードにします．そのとき，対象キーの音名スペースが黒く反転します．あとは，そのスペースに6文字以内で音名を入力し，CRを押せば完了です．

実際にサウンドを作るにはエンベロープと周波数の設定が必要です．

エンベロープで音量の強弱を表現し，周波数では音階つまり音の高低を表現します．この周波数シフトの設定によってゲームなどに使われている効果音や電話の呼び出し音なども表現できます．まずエンベロープのつくり方ですが，ためしにピアノのエンベロープを設定してみましょう．

①左下に始点カーソルをもってきて，〔F 1〕キーを押します．これで始点が設定されます（図4・16）．

②終点カーソルを右にひとつ移動させ，さらに最上部まで移動させます．そして，〔F 2〕キーを押します．このとき，カーソルを移動させるのには，カーソルキーを用いますが，SHIFTキーといっしょに押すと，8スペース分移動します（図4・17）．

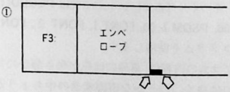
③次にまた，始点と終点を決めて，エンベロープを分割した直線を入力していきます（図4・18）．

次に，周波数シフトの設定ですが，まず，〔F 3〕キーを押して，このブロックの左端を白く反転させます（図4・19参照）．

これでこのブロックに入力が可能となります．たて軸は，−16から+16までありますが，この1めもりは音階で半音階を表わします．また，横軸はエンベロープと同じく時間経過を表わします．

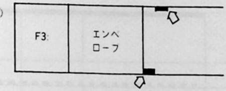
効果音をつくってみましょう

エンベロープは，初期設定のまま（ひとつめの



始点カーソルは F1 キーで確定される．

図4・16 始点の設定



始点カーソルと終点カーソルの間のエンベロープが F2 キーを押すことによって確定

図4・17 終点の設定

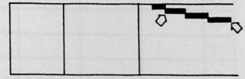


図4・18 エンベロープの確定

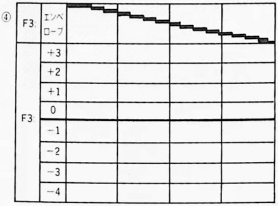


図4・19　エンベロープの例

ラバーバンドは最下部に，そして次から最後まで最上部に設定)で，周波数を1スペースずつ0，＋4，＋7と交互に設定していきます．始点・終点カーソルの移動と設定はエンベロープでしたのと同じやり方です（図4・20参照）．

0をドとすると＋4はミ，＋7はソになっています．できあがったら音名キーを押してみましょう．

4-3-3 実際の作成例

さて，最後にこれまでにつくってきたデータを，実際に使用してみることにしましょう．

“DEMOM 1.00”と BASIC を使って，楽しいアニメーションをつくってみましょう.ここでは PSGEDIT, PSG 1.00, PSGM 1.00, FONT 1, FONT 2, FONT 3のプログラムを使用します．

プログラムの内容は「草原には花が咲き静かに川は流れ山は白く，のどかな田園風景の中ちょうがひらひらと，ときおり牛がユーモラスに鳴き……」

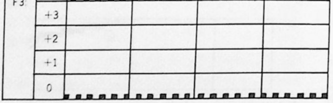


図4・20
周波数シフト

牛の前部〈PCG65〉

4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
4 4 4 1 1 4 4 4	4 4 4 1 1 4 4 4	4 4 4 4 4 4 4 4
4 4 1 1 4 4 4 4	4 4 4 4 1 1 4 4	4 4 4 4 4 4 4 4
4 4 1 1 4 4 4 4	4 4 4 4 1 1 4 4	4 4 4 4 4 4 4 4
4 4 4 1 1 1 3 3	3 3 1 1 1 4 4 4	4 4 4 4 4 4 4 7
7 7 7 7 7 7 7 7	7 7 7 7 7 7 7 0	7 7 7 7 7 7 7 0
0 7 7 7 7 7 7 7	7 7 7 7 7 7 0 7	7 7 7 0 0 0 0 0
4 0 7 0 7 7 7 7	7 7 7 7 7 0 7 7	7 7 7 7 7 0 0 0
4 4 0 7 7 2 7 7	2 7 7 0 0 7 7 0	0 0 7 7 7 7 7 7
4 4 7 7 7 7 7 7	7 7 7 0 7 7 0 0	0 7 7 7 7 7 7 7
4 7 7 7 7 7 7 7	7 7 7 0 7 0 0 0	7 7 7 7 7 7 7 7
4 7 7 7 7 7 7 7	7 7 7 0 4 7 7 7	7 7 7 7 7 7 0 0
4 7 7 7 7 7 7 7	7 0 7 0 4 7 0 0	7 7 7 7 7 0 0 7
4 4 7 7 0 0 0 0	0 7 0 4 4 4 7 0	0 7 7 7 7 7 7 7
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 7	7 7 7 0 7 7 7 7
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 7	7 7 7 0 7 7 7 7
4 4 4 4 4 4 4 4	4 4 4 4 4 4 7 0	7 7 7 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 7 0	7 7 0 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 7	7 7 0 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	7 7 7 1 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 1	7 7 1 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 1 1	1 1 4 4 4 4 4 4

牛の後部〈PCG68〉

4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4	4 4 4 4 4 4 4 4
7 7 7 7 7 7 7 7	7 7 7 7 4 4 4 4	4 4 4 4 4 4 4 4
0 7 7 7 7 0 0 0	0 0 0 7 7 7 4 4	4 4 4 4 4 4 4 4
0 0 7 7 7 7 7 0	0 0 0 0 0 7 7 7	4 4 4 4 4 4 4 4
0 7 7 7 7 7 7 7	7 7 7 7 7 7 7 7	7 4 4 4 4 4 4 4
7 7 7 7 7 7 7 7	7 7 7 7 7 7 7 7	4 7 4 4 4 4 4 4
7 7 7 7 7 7 7 7	0 0 0 0 0 7 7 7	4 4 7 4 4 4 4 4
0 0 7 7 7 7 7 0	0 0 0 0 7 7 7 7	4 4 7 4 4 4 4 4
7 0 7 7 7 7 7 7	7 7 7 7 7 7 7 7	4 4 7 4 4 4 4 4
7 7 7 7 0 0 7 7	7 7 7 7 7 7 7 4	4 4 7 4 4 4 4 4
7 7 7 7 7 7 0 7	7 7 7 7 7 7 7 4	4 4 7 4 4 4 4 4
7 7 7 7 7 7 7 0	7 7 7 7 7 7 4 4	4 4 3 4 4 4 4 4
7 7 7 7 7 7 7 7	0 7 7 7 7 4 4 4	4 4 3 4 4 4 4 4
4 7 7 7 7 7 4 4	7 7 7 7 7 4 4 4	4 4 4 4 4 4 4 4
4 3 4 3 4 3 4 4	7 0 7 7 7 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 7 0 7 7 7 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 7 0 7 7 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	1 0 1 7 7 4 4 4	4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4	4 1 1 1 1 4 4 4	4 4 4 4 4 4 4 4

図4・21　フォントサンプル

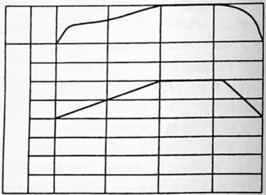


図4・22　牛の鳴きごえのエンベロープ

といったものです．パターンエースでつくったデータに牛のデータを追加し，ちょうを舞わせながらキー入力で牛を鳴かせます．

DEMOプログラム用に新しいQDを用意し，DEMO 1.00をコピーしておいて下さい．コピーはTRANSを使用して下さい．データを追加する時に入力用QDは，今までに作ったデータのQDより出力用QDは，用意したQDを使って下さい．

○牛のフォントを作る

フォントサンプル（図4・21）を参照して，オリジナルキャラクタを作って下さい．サイズ24の牛の前部と後部，頭部の差し換え用の計3つです．

○牛の鳴声をつくる

ミュージックエースでつくったサウンドに追加します．図4・22のようなエンベロープと周波数シフトをKキーの音として登録しておきます．データをつくっているときに聞く音は，音程が高いので牛の声には聞こえませんが，実際には音程を下げますので心配いりません．ファイル名を"MYSOUND"としてセーブしておきます．

デモプログラムを作る

リスト4・4を見ながら，タイプミスに注意してプログラムを打ち込みます．できあがったプログラムを走らせる前に，DEMOM 1.00，フォントデータ，マップデータ，楽譜データといっしょに1枚のQDにセーブしておきましょう．そして，そのQDをセットしてから，次の手順でRUNです．BGMに関係なく，Zキーを押せば，牛が「モーッ」と鳴きます．

```
NEWON 2
LOAD  "DEMO"
RUN      └─いまつくったデモプログラムです．
```

さあ，うまくいきましたか？このプログラムを参考にして，あなたもオリジナルプログラムを作ってください．

```
10 LIMIT$BFFF:LOAD"DEMO1.00"
20 USR($C000,"MYFONT"):LOAD"MYMAP":LOAD"MYGAKUFU":LOAD"SOUND"
30 INIT"CRT:I":CLS3
40 POKE$C060,$FF:POKE$C061,$FE:POKE$C062,$28:POKE$C063,0:POKE$
C064,0:POKE$C065,$C8:POKE$C066,$28:POKE$C067,$18
50 USR($C006)
60 POKE$C06D,4
70 POKE$C06F,10:POKE$C070,15
80 X1=120:Y1=110:K=0
100 USR($C009)
110 USR($C012)
120 J=K:S=X1:T=0:U=Y1:GOSUB250
130 USR($C00C)
140 USR($C009)
150 USR($C00F)
160 USR($C01B):IFPEEK($C071)=$5A GOSUB300
170 IFK=0THENK=2:ELSEK=0
180 A=INT(RND(1)*30)-15:B=INT(RND(1)*20)-10
190 X1=X1+A:Y1=Y1+B
200 IFX1<0THENX1=0
210 IFX1>255THENX1=255
220 IFY1<0THENY1=0
230 IFY1>183THENY1=183
240 GOTO120
250 POKE$C068,0:POKE$C069,J:POKE$C06A,S:POKE$C06B,T:POKE$C06C,
U:RETURN
300 J=13:S=200:T=0:U=120:GOSUB250
310 USR($C00C)
320 USR($C009)
330 USR($C00F)
340 USR($C018):WAIT1280:RETURN
```

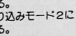
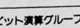
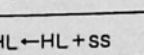
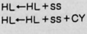
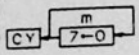
リスト4・4　デモプログラム

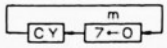
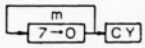
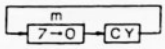
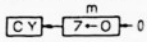
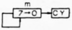
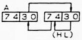
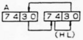
APPENDIX——1：Z80命令表

ニーモニック	実行内容	命令コード 76 543 210
8ビットロードグループ		
LD r, r'	r←r'	01 r r'
LD r, n	r←n	00 r 110 ← n →
LD r, (HL)	r←(HL)	01 r 110
LD r, (IX+d)	r←(IX+d)	11 011 101 01 r 110 ← d →
LD r, (IY+d)	r←(IY+d)	11 111 101 01 r 110 ← d →
LD (HL), r	(HL)←r	01 110 r
LD (IX+d), r	(IX+d)←r	11 011 101 01 110 r ← d →
LD (IY+d), r	(IY+d)←r	11 111 101 01 110 r ← d →
LD (HL), n	(HL)←n	00 110 110 ← n →
LD (IX+d), n	(IX+d)←n	11 011 101 00 110 110 ← d → ← n →
LD (IY+d), n	(IY+d)←n	11 111 101 00 110 110 ← d → ← n →
LD A, (BC)	A←(BC)	00 001 010
LD A, (DE)	A←(DE)	00 011 010
LD A, (nn)	A←(nn)	00 111 010 ← n → ← n →
LD (BC), A	(BC)←A	00 000 010
LD (DE), A	(DE)←A	00 010 010
LD (nn), A	(nn)←A	00 110 010 ← n → ← n →
LD A, I	A←I	11 101 101 01 010 111
LD A, R	A←R	11 101 101 01 011 111
LD I, A	I←A	11 101 101 01 000 111
LD R, A	R←A	11 101 101 01 001 111
16ビットロードグループ		
LD dd, nn	dd←nn	00 dd0 001 ← n → ← n →
LD IX, nn	IX←nn	11 011 101 00 100 001 ← n → ← n →
LD IY, nn	IY←nn	11 111 101 00 100 001 ← n → ← n →

ニーモニック	実行内容	命令コード 76 543 210
LD HL, (nn)	H←(nn+1) L←(nn)	00 101 010 ← n → ← n →
LD dd, (nn)	dd_H←(nn+1) dd_L←(nn)	11 101 101 01 dd1 011 ← n → ← n →
LD IX, (nn)	IX_H←(nn+1) IX_L←(nn)	11 011 101 00 101 010 ← n → ← n →
LD IY, (nn)	IY_H←(nn+1) IY_L←(nn)	11 111 101 00 101 010 ← n → ← n →
LD (nn), HL	(nn+1)←H (nn)←L	00 100 010 ← n → ← n →
LD (nn), dd	(nn+1)←dd_H (nn)←dd_L	11 101 101 01 dd0 011 ← n → ← n →
LD (nn), IX	(nn+1)←IX_H (nn)←IX_L	11 011 101 00 100 010 ← n → ← n →
LD (nn), IY	(nn+1)←IY_H (nn)←IY_L	11 111 101 00 100 010 ← n → ← n →
LD SP, HL	SP←HL	11 111 001
LD SP, IX	SP←IX	11 011 101 11 111 001
LD SP, IY	SP←IY	11 111 101 11 111 001
PUSH qq	(SP−2)←qq_L (SP−1)←qq_H	11 qq0 101
PUSH IX	(SP−2)←IX_L (SP−1)←IX_H	11 011 101 11 100 101
PUSH IY	(SP−2)←IY_L (SP−1)←IY_H	11 111 101 11 100 101
POP qq	qq_H←(SP+1) qq_L←(SP)	11 qq0 001
POP IX	IX_H←(SP+1) IX_L←(SP)	11 011 101 11 100 001
POP IY	IY_H←(SP+1) IY_L←(SP)	11 111 101 11 100 001
エクスチェンジグループ、ブロック転送及び、サーチグループ		
EX DE, HL	DE↔HL	11 101 011
EX AF, AF'	AF↔AF'	00 001 000
EXX	(BC)↔(BC') (DE)↔(DE') (HL)↔(HL')	11 011 001
EX (SP), HL	H↔(SP+1) L↔(SP)	11 100 011
EX (SP), IX	IX_H↔(SP+1) IX_L↔(SP)	11 011 101 11 100 011
EX (SP), IY	IY_H↔(SP+1) IY_L↔(SP)	11 111 101 11 100 011

ニーモニック	実行内容	命令コード 76 543 210
LDI	(DE)←(HL)	11 101 101
	DE←DE+1	10 100 000
	HL←HL+1	
	BC←BC−1	
LDIR	(DE)←(HL)	11 101 101
	DE←DE+1	10 110 000
	HL←HL+1	
	BC←BC−1	
	BC=0までくりかえす	
LDD	(DE)←(HL)	11 101 101
	DE←DE−1	10 101 000
	HL←HL−1	
	BC←BC−1	
LDDR	(DE)←(HL)	11 101 101
	DE←DE−1	10 111 000
	HL←HL−1	
	BC←BC−1	
	BC=0までくりかえす	
CPI	A−(HL)	11 101 101
	HL←HL+1	10 100 001
	BC←BC−1	
CPIR	A−(HL)	11 101 101
	HL←HL+1	10 110 001
	BC←BC−1	
	A=(HL)又はBC=0までくりかえす	
CPD	A−(HL)	11 101 101
	HL←HL−1	10 101 001
	BC←BC−1	
CPDR	A−(HL)	11 101 001
	HL←HL−1	10 111 001
	BC←BC−1	
	A=(HL)又はBC=0までくりかえす	
8ビット演算・論理グループ		
ADD A, r	A←A+r	10 [000] r
AD A, n	A←A+n	11 [000] 110
		← n →
ADD A, (HL)	A←A+(HL)	10 [000] 110
ADD A, (IX+d)	A←A, (IX+d)	11 011 101
		10 [000] 110
		← d →
ADD A, (IY+d)	A←A+(IY+d)	11 111 101
		10 [000] 110
		← d →
ADC A, s	A←A+s+CY	[001]
SUB s	A←A−s	[010]
SBC A, s	A←A−s−CY	[011]
AND s	A=A∧s	[100]
OR s	A←A∨s	[110]
XOR s	A←A⊕s	[101]
CP s	A−s	[111]
INC r	r←r+1	00 r [100]
INC (HL)	(HL)←(HL)+1	00 110 [100]
INC (IX+d)	(IX+d)	11 011 101
	←(IX+d)+1	00 110 [100]
		← d →
INC (IY+d)	(IY+d)	11 111 101
	←(IY+d)+1	00 110 [100]

ニーモニック	実行内容	命令コード 76 543 210
		← d →
DEC m	m←m−1	[101]
一般目的の演算、及びCPUコントロールグループ		
DAA	加減算後のAの内容の10進補正を行う。	00 100 111
CPL	A←$\overline{A}$	00 101 111
NEG	A←$\overline{A}$+1	11 101 101
		01 000 100
CCF	CY←$\overline{CY}$	00 111 111
SCF	CY←1	00 110 111
NOP	何も実行されないがPCの内容はインクリメントされる	00 000 000
HALT	CPU停止	01 110 110
DI	IFF←0	11 110 011
EI	IFF←1	11 111 011
IM0	割り込みモード0にする。	11 101 101
		01 000 110
IM1	割り込みモード1にする。	11 101 101
		01 010 110
IM2	割り込みモード2にする。	11 101 101
		01 011 110
16ビット演算グループ		
ADD HL, ss	HL←HL+ss	00 ss1 001
ADC HL, ss	HL←HL+ss+CY	11 101 101
		01 ss1 010
SBC HL, ss	HL←HL−ss−CY	11 101 101
		01 ss0 010
ADD IX, pp	IX←IX+pp	11 011 101
		00 pp1 001
ADD IY, rr	IY←IY+rr	11 111 101
		00 rr1 001
INC ss	ss←ss+1	00 ss0 011
INC IX	IX←IX+1	11 011 101
		00 100 011
INC IY	IY←IY+1	11 111 101
		00 100 011
DEC ss	ss←ss−1	00 ss1 011
DEC IX	IX←IX−1	11 011 101
		00 101 011
DEC IY	IY←IY−1	11 111 101
		00 101 011
ローテイト・シフトグループ		
RLCA		00 000 111
RLA		00 010 111
RRCA		00 001 111
RRA		00 011 111
RLC r		11 001 011
		00 [000] r
RLC (HL)		11 001 011
		00 [000] 110

ニーモニック	実行内容	命令コード 76 543 210
RLC (IX+d)		11 011 101 11 001 011 ← d → 00 [000] 110
RLC (IY+d)		11 111 011 11 001 011 ← d → 00 [000] 110
RL m		[010]
RRC m		[001]
RR m		[011]
SLA m		[100]
SRA m		[101]
SRL m		[111]
RLD		11 101 101 01 101 111
RRD		11 101 101 01 100 111
ビットセット、リセット及びテストグループ		
BIT b, r	$Z \leftarrow \overline{r_b}$	11 001 011 01 b r
BIT b, (HL)	$Z \leftarrow \overline{(HL)_b}$	11 011 011 01 b 110
BIT b, (IX+d)	$Z \leftarrow \overline{(IX+d)_b}$	11 011 101 11 001 011 ← d → 01 b 110
BIT b, (IY+d)	$Z \leftarrow \overline{(IY+d)_b}$	11 111 101 11 001 011 ← d → 01 b 110
SET b, r	$r_b \leftarrow 1$	11 001 011 [11] b r
SET b, (HL)	$(HL)_b \leftarrow 1$	11 001 011 [11] b 110
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	11 001 101 11 001 011 ← d → [11] b 110
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	11 111 101 11 001 011 ← d → [11] b 110
RES b, m	$m_b \leftarrow 0$	[10]

ニーモニック	実行内容	命令コード 76 543 210
ジャンプグループ		
JP nn	PC←nn	11 000 011 ← n → ← n →
JP cc, nn	ccが真のとき、 PC←nn	11 cc 010 ← n → ← n →
JR e	PC←PC+e	00 011 000 ← e-2 →
JR C, e	C=0のとき、 次の命令へ C=1のとき、 PC←PC+e	00 111 000 ← e-2 →
JR Z, e	Z=0のとき、 次の命令へ Z=1のとき、 PC←PC+e	00 101 000 ← e-2 →
JR NC, e	C=1のとき、 次の命令へ C=0のとき、 PC←PC+e	00 110 000 ← e-2 →
JR NZ, e	Z=1のとき、 次の命令へ Z=0のとき、 PC←PC+e	00 100 000 ← e-2 →
JP (HL)	PC←HL	11 101 001
JP (IX)	PC←IX	11 011 101 11 101 001
JP (IY)	PC←IY	11 111 101 11 101 001
DJNZ e	B←B−1 B=0のとき、 次の命令へ B≠0のとき、 PC←PC+e	00 010 000 ← e-2 →
コール、リターングループ		
CALL nn	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ PC←nn	11 001 101 ← n → ← n →
CALL cc, nn	ccが真のとき、 CALL nnと同じ ccが偽のとき継続	11 cc 100 ← n → ← n →
RET	$PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP+1)$	11 001 001
RET cc	ccが真のとき、 RETと同じ ccが偽なら継続	11 cc 000
RETI	割り込みからの リターン	11 101 101 01 001 101
RETN	NMIからのリターン	11 101 101 01 000 101
RST p	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ $PC_H \leftarrow 0$ $PC_L \leftarrow p$	11 t 111

ニーモニック	実行内容	命令コード 76 543 210
入力・出力グループ		
IN A, (n)	A←(n)	11 011 011 ← n →
IN r, (C)	r←(C)	11 101 101 01 r 000
INI	(HL)←(C) B←B−1 HL←HL+1	11 101 101 10 100 010
INIR	(HL)←(C) B←B−1 HL←HL+1 B=0までくりかえす	11 101 101 10 110 010
IND	(HL)←(C) B←B−1 HL←HL−1	11 101 101 10 101 010
INDR	(HL)←(C) B←B−1 HL←HL−1 B=0までくりかえす	11 101 101 10 111 010

ニーモニック	実行内容	命令コード 76 543 210
OUT (n), A	(n)←A	11 010 011 ← n →
OUT (C), r	(C)←r	11 101 101 01 r 001
OUTI	(C)←(HL) B←B−1 HL←HL+1	11 101 101 10 100 011
OTIR	(C)←(HL) B←B−1 HL←HL+1 B=0までくりかえす	11 101 101 10 110 011
OUTD	(C)←(HL) B←B−1 HL←HL−1	11 101 101 10 101 011
OTDR	(C)←(HL) B←B−1 HL←HL−1 B=0までくりかえす	11 101 101 10 111 011

(注) 前記命令表で使用してある記号の意味は次のとおりです。

r, r′	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

dd, ss	ペアレジスタ
00	BC
01	DE
10	HL
11	SP

qq	ペアレジスタ
00	BC
01	DE
10	HL
11	AF

pp	ペアレジスタ
00	BC
01	DE
10	IX
11	SP

rr	ペアレジスタ
00	BC
01	DE
10	IY
11	SP

∧：論理積
∨：論理和
⊕：排他的論理和

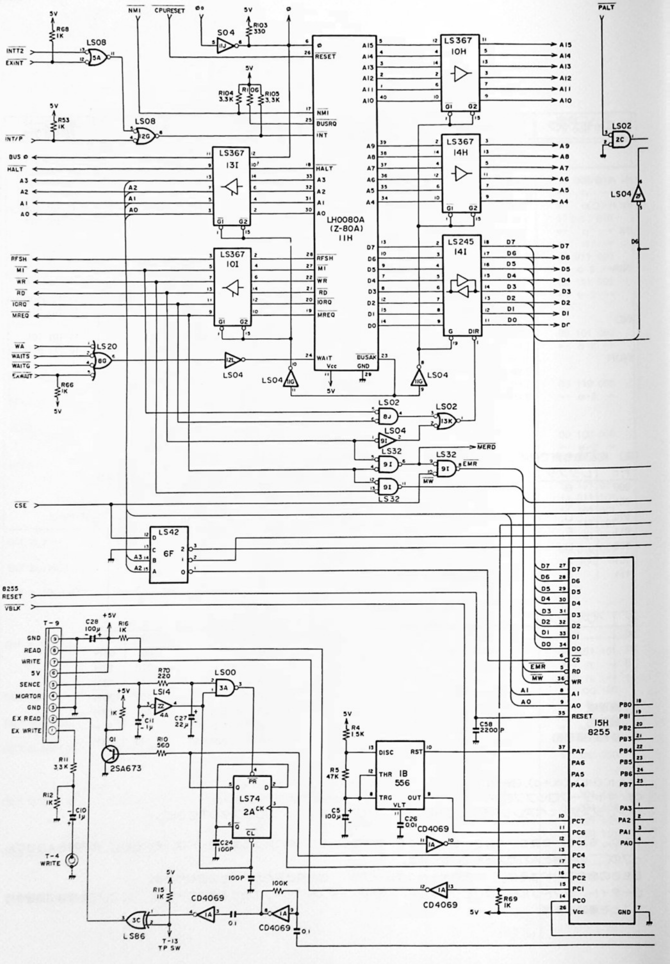
b	ビットセット
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

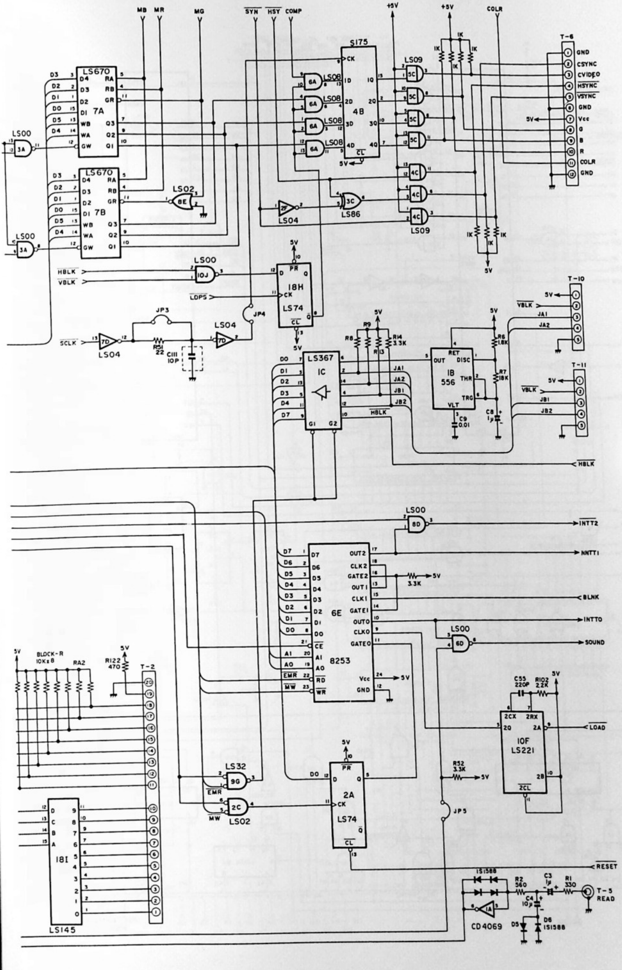
cc	コンディション	
000	NZ	non zero
001	Z	zero
010	NC	non carry
011	C	carry
100	PO	parity odd
101	PE	parity even
110	P	sign positive
111	M	sign negative

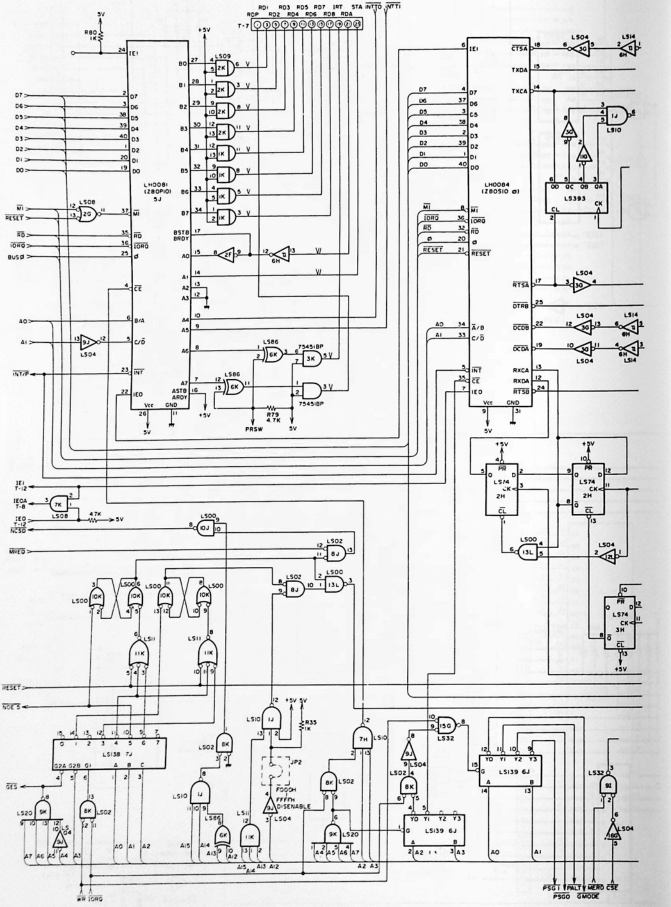
t	p
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

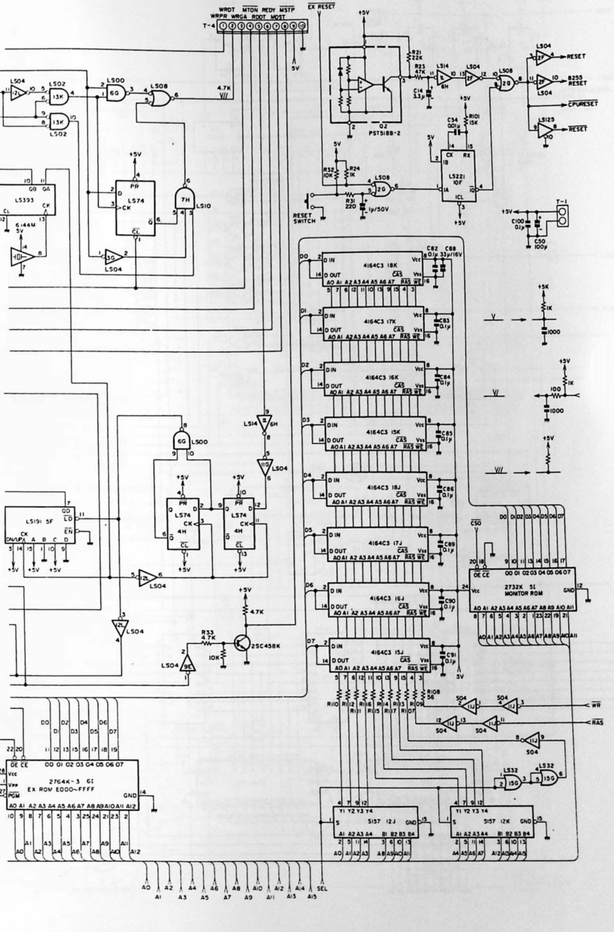
s：r, n, (HL), (IX+d), (IY+d)
m ：r, (HL), (IX+d), (IY+d)
CY：キャリーフロップフロップ
m_b：ビットb又はロケーションm
(ペアレジスタ)$_H$：ペアレジスタの上位8ビット
(ペアレジスタ)$_L$：ペアレジスタの下位8ビット

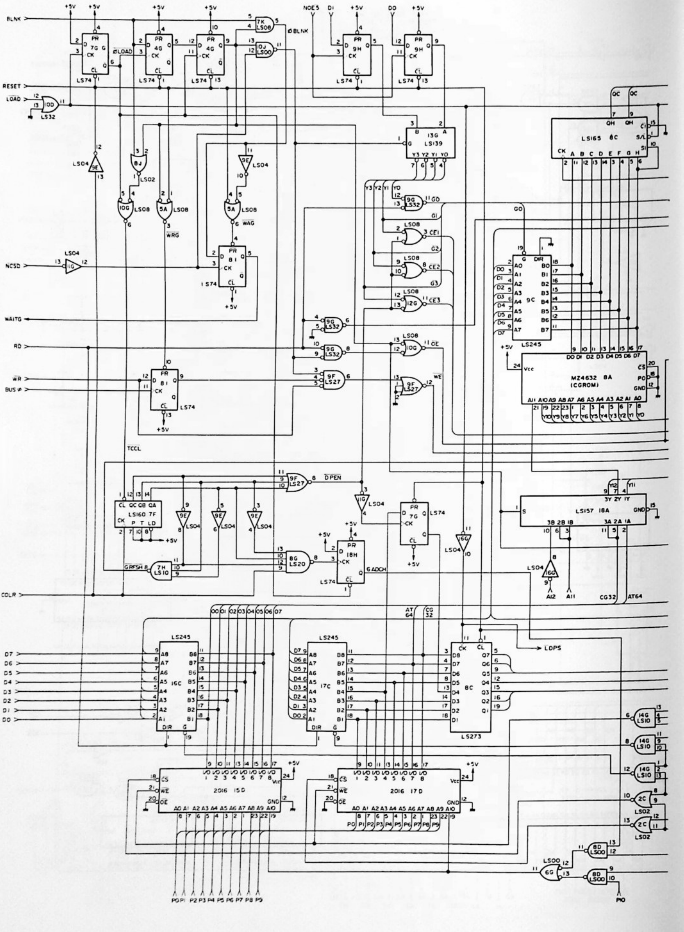
ADC、SUB、SBC、AND、OR、XOR、CPのニーモニックに対する命令コードは、それらの□の内容をADDグループの□の内容と入れかえたものになります。
DECの命令コードはその□の内容をINCグループの□の内容と入れかえたものになります。
ローテイト、シフトグループ及びビットセット、リセット、テストグループにおける命令コードの□についても同様の処理を行うことを意味します。

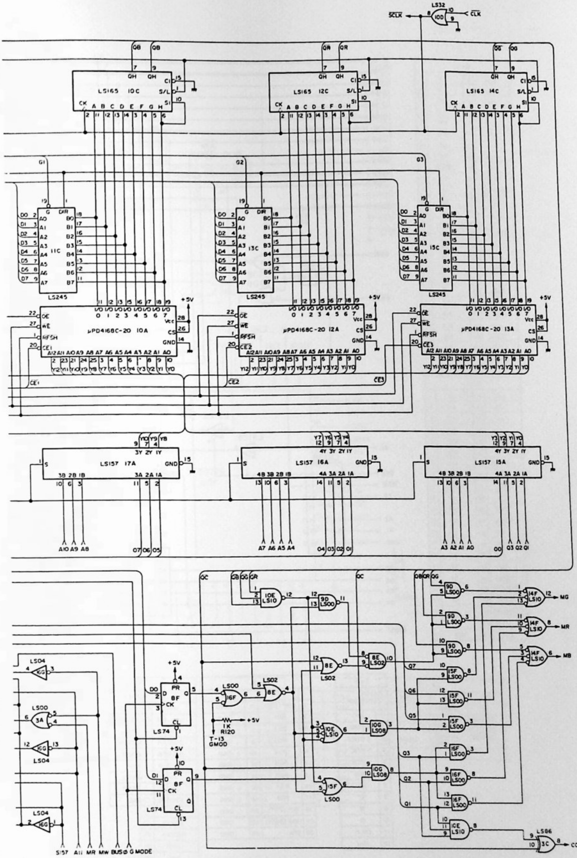

LH0080A
(Z-80A)
11H
LS367
10H
14H
13I
10I
LS245
14I
LS08
LS02
LS04
LS32
LS20
LS42
6F
LS00
LS14
LS74
2A
556
1B
CD4069
LS86
15H
8255
2SA673
T-9
T-4
WRITE
T-13
TP SW
INTT2
EXINT
INT/P
BUS Φ
HALT
RFSH
MI
WR
RD
IORQ
MREQ
WA
WAITS
WAITG
EXWAIT
CSE
8255 RESET
VBLK
NMI
CPURESET
PALT
MERD
EMR
MW
GND
READ
WRITE
5V
SENCE
MORTOR
EX READ
EX WRITE


MB
MR
MG
SYN
HSY
COMP
+5V
+5V
COLR
LS670
7A
LS670
7B
LS00
3A
LS00
3A
LS02
8E
S175
4B
LS08
6A
LS09
5C
4C
LS86
3C
LS04
2F
LS09
T-6
GND
CSYNC
CVIDEO
HSYNC
VSYNC
GND
Vcc
G
B
R
COLR
GND
1K
5V
LS00
10J
HBLK
VBLK
LDPS
18H
LS74
JP4
JP3
R51
22
C111
10P
SCLK
LS04
7D
LS04
7D
LS367
1C
R8
R9
R13
R14
3.3K
JA1
JA2
JB1
JB2
RET
OUT
DISC
THR
TRG
VLT
1B
556
R6
1.8K
R7
18K
C8
1μ
C9
0.01
T-10
T-11
LS00
8D
INTT2
NNTT1
8253
6E
OUT2
CLK2
GATE2
OUT1
CLK1
GATE1
OUT0
CLK0
GATE0
Vcc
GND
CE
RD
WR
EMR
MW
BLNK
INTT0
SOUND
LS00
6D
C55
220P
R102
2.2K
2CX
2RX
2Q
2A
2B
2CL
10F
LS221
LOAD
R52
3.3K
JP5
BLOCK-R
10K×8
RA2
R122
470
T-2
LS32
9G
LS02
2C
2A
LS74
PR
CL
181
LS145
RESET
1S1588
R2
560
C3
1μ
R1
330
T-5
READ
C4
10μ
D5
D6
1S1588
1A
CD4069

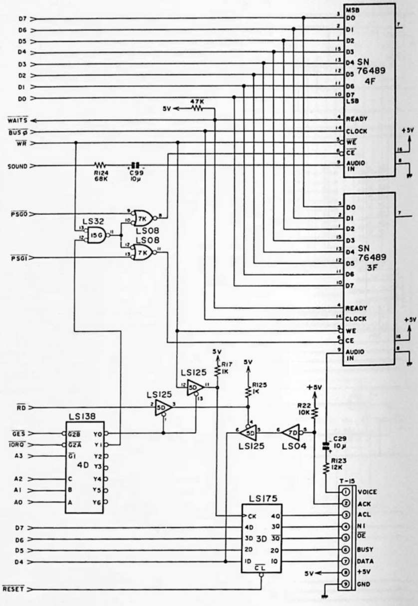

LH0081
(Z80PIO)
5J
LH0084
(Z80SIO Ø)
LS393
LS74
2H
3H
LS138 7J
LS139 6J
75451BP
R80
1K
R79
4.7K
R35
1K
JP2
F000H
FFFFH
DISENABLE
PRSW
T-7
RDP
RD1
RD2
RD3
RD4
RD5
RD6
RD7
RD8
IRT
RDA
STA
INTT0
INTT1
CTSA
TXDA
TXCA
RTSA
DTRB
DCDB
DCDA
RXCA
RXDA
RTSB
IE1
T-12
IE0A
T-8
IE0
NCSD
MREQ
RESET
NOE 5
GES
INT/P
BUSØ
WR
IORQ
PSG1
PSG0
PALT
GMODE
MERO
CSE


T-4
WRDT MTON REDY MSTP
WRPR WRGA RDDT MDST
EX RESET
+5V
R21 22K
R23 47K
C14 3.3μ
Q2 PST518B-2
LS14 6H
LS04 2F
LS08 2G
RESET
8255 RESET
CPURESET
LS125
R101 15K
C34 0.01μ
LS221 10F
R32 10K
R24 1K
R31 220
1μ/50V
RESET SWITCH
T-1
C100 0.1μ
C50 100μ
LS04 12L
LS02 13K
LS00 6G
LS08 7H
LS393
6.144M
LS74
LS10
LS04 3G
4.7K
4164C3 18K
4164C3 17K
4164C3 16K
4164C3 15K
4164C3 18J
4164C3 17J
4164C3 16J
4164C3 15J
C82 0.1μ
C88 33μ/16V
C83 0.1μ
C84 0.1μ
C85 0.1μ
C86 0.1μ
C89 0.1μ
C90 0.1μ
C91 0.1μ
2732K 5I MONITOR ROM
LS191 5F
LS74 4H
LS14 6H
LS04 11G
R33 4.7K
10K
2SC458K
LS04 9E
LS04 12L
2764K-3 6I
EX ROM E000~FFFF
R108 56
R110 R112 R116 R114 R113 R109
R111 R115 R117 R107
S04 11J
WR
RAS
LS32 15G
S157 12J
S157 12K
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15 SEL


BLNK
RESET
LOAD
LS32
LS74
LS04
LS08
LS02
LS00
ØLOAD
ØBLNK
NOE5
WRG
WAG
NCSD
WAITG
RD
WR
BUS≠
LS139
LS165 8C
LS245
LS27
LS20
LS10
LS160 7F
MZ4632 8A
(CGROM)
LS157 18A
TCCL
GRFSH
DPEN
6ADCH
COLR
CE1
CE2
CE3
OE
WE
A12
A11
CG32
AT64
LDPS
LS273
16C
17C
2016 15D
2016 17D
P0 P1 P2 P3 P4 P5 P6 P7 P8 P9
P10



LS32
10D
SCLK
CLK
LS165 10C
LS165 12C
LS165 14C
QH QH
CK A B C D E F G H
C1 S/L SI
QB QB
QR QR
QG QG
G1
G2
G3
LS245
11C
13C
15C
G DIR
D0 D1 D2 D3 D4 D5 D6 D7
A0 A1 A2 A3 A4 A5 A6 A7
B0 B1 B2 B3 B4 B5 B6 B7
μPD4168C-20 10A
μPD4168C-20 12A
μPD4168C-20 13A
I/O 0 1 2 3 4 5 6 7
OE WE RFSH CE1
CE2
CE3
Vcc CS GND
+5V
A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
Y12 Y11 Y10 Y9 Y8 Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0
LS157 17A
LS157 16A
LS157 15A
S
4Y 3Y 2Y 1Y
4B 3B 2B 1B
4A 3A 2A 1A
A10 A9 A8
O7 O6 O5
A7 A6 A5 A4
O4 O3 O2 O1
A3 A2 A1 A0
O0 Q3 Q2 Q1
QC
QB QG QR
10E LS10
9D LS00
8E LS02
14F LS10
MG
MR
MB
9D LS00
15F LS00
16F LS00
10G LS08
10E LS10
LS04
16G
LS00
3A
LS74
8F
PR D CK Q CL
D0
D1
16F
1K R120
T-13 GMOD
Q7 Q6 Q5 Q3 Q2 Q1
LS86
3C
COMP
S157 A11 MR MW BUSØ G MODE

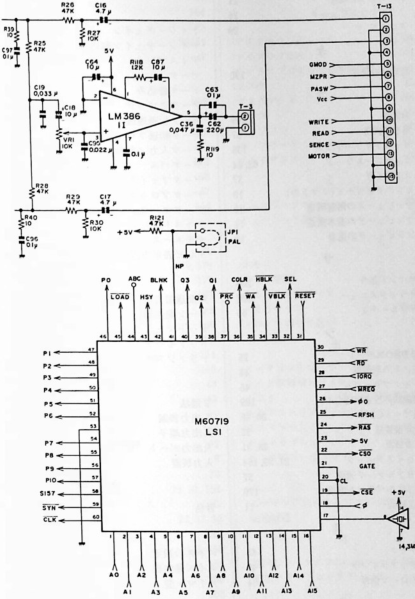


T-6	
CCLOR ENCORDER	
1	GND
2	CSYNC
3	CVIDEO
4	H SYNC
5	VCYNC
6	GND
7	Vcc
8	G
9	R
10	B
11	COLR
12	GND

T-7			
	A		B
1	RDP (STB)	2	GND
3	RD1	4	GND
5	RD2	6	GND
7	RD3	8	GND
9	RD4	10	GND
11	RD5	12	GND
13	RD6	14	GND
15	RD7	16	GND
17	RD8	18	GND
19	IRT (RST)	20	GND
21	RDA (BUSY)	22	GND
23	STA	24	GND

T-9	
	CASET
1	EX WRITE
2	EX READ
3	GND
4	MOTOR
5	SENCE
6	+5V
7	WRITE
8	READ
9	GND

T-10	
	JOYSTICK B
1	Vcc
2	V BLNK
3	JB1
4	JB2
5	GND



T-8			
1	D0	2	GND
3	D1	4	GND
5	D2	6	GND
7	D3	8	GND
9	D4	10	GND
11	D5	12	GND
13	D6	14	GND
15	D7	16	GND
17	BUSφ	18	GND
19	A0	20	RESET
21	A1	22	EX RESET
23	A2	24	HALT
25	A3	26	IEOA
27	A4	28	M1
29	A5	30	EX WAIT
31	A6	32	WR
33	A7	34	GND
35	A8	36	RD
37	A9	38	GND
39	A10	40	IORQ
41	A11	42	GND
43	A12	44	MERQ
45	A13	46	GND
47	A14	48	EXINT
49	A15	50	NMI

T-12			
1	Vcc	2	Vcc
3	D2	4	D3
5	D1	6	D4
7	D0	8	D5
9	GND	10	D6
11	A15	12	D7
13	A14	14	BUSφ
15	A13	16	M1
17	A12	18	WR
19	A11	20	RD
21	A10	22	IORQ
23	A9	24	MREQ
25	A8	26	GND
27	A7	28	HALT
29	A6	30	IEI
31	A5	32	IEO
33	A4	34	RESET
35	A3	36	EX RESET
37	A2	38	INT/P
39	A1	40	EX WAIT
41	A0	42	NMI
43	GND	44	GND

T-11	
	JOYSTICK A
1	Vcc
2	V BLNK
3	JA 1
4	JA 2
5	GND

T-13	
1	L-CH
2	GND
3	R-CH
4	GND
5	G MOD
6	PRSW
7	TPSW
8	Vcc
9	GND
10	WRITE
11	READ
12	SENCE
13	MOTOR
14	GND
15	GND

T-14	
	QD
1	WRPR
2	WRDT
3	WRGA
4	MT ON
5	RDDT
6	REDY
7	MDST
8	MSTP
9	Vcc
10	GND

T-15	
	VOICE
1	VOICE
2	ACK
3	ACL
4	N 1
5	O E
6	BUSY
7	DATA
8	Vcc
9	GND

DIP SW			
NO	NAME	ON	OFF
1	TP SW	NORMAL	INV
2	PR SW	MZ PRINTER	CENTER PRINTER
3	GMOD	GRAPHIC NON	GRAPHIC
4			

用語総索引

50音順，ＡＢＣ順

ア

アーリーライト 157
アッテネーション(ミュージック) 194
アッテネーションの設定 190
アトリビュートエリア 154
アドレスバス 47, 48
アプリケーションプログラム 35, 37

イ

イニシャライズ(クイックディスクの) 166
イニシャルプログラムローダ 37
色情報 164
インターフェイス 22
インタープリタ 37
インデックスレジスタ 47
インフォメーションブロック 166, 179

エ

エンベロープ 192, 208

オ

大きな絵 203
音の強弱 194
音の高低 194
音の種類 208
音の鳴り方 194
音を構成する要素 194
オペレーティングシステム 11, 35, 36
音楽機能 21
音声ボード 32
音符データ 193

カ

カーソルキー 202
外部記憶装置 12, 28
書込み禁止 168
書込みシーケンス 155
拡張I/Oバス 23
カセットインターフェイス 180
カセットテープのデータフォーマット 181
カセットの制御 153
カラーパレット 154
漢字ROMボード 22, 25

キ

キーの制御 153
キーボード 30
キーボードスキャン 181
キーマトリクス 181
記憶装置 12, 29, 166
機械語 29
機械語サブルーチン(PCG) 197, 198
機械語プログラムの書込み 197
キャラクタ画面 154
キャラクタジェネレータ 154
9Z-502M 174

ク

クイックディスク 18, 19, 32, 152
グラフィック機能 21
クリーンコンピュータ 26
クロック 29

ケ

減衰量の設定 190

コ

効果音 209
高水準言語 37
コーリングシーケンス 139
コントロールワード 62, 64
コンパイラ 37
コンパチビリティ(ソフトの) 18
コンピュータの階層構造 12
コンピュータの基本構造 11
コンピュータの進歩 10

サ

サウンド作り 208
サウンドメニュー 208
サブルーチン 45

シ

辞書ROMボード 25
システム制御線 48
ジャンプテーブル 45
周波数の設定(PSGの) 189
16ビットCPU 20, 46
主記憶装置 32
出力装置 28, 31
ジョイスティック 21, 30, 154
シリアルデータ通信 57
シリアルパラレル変換 170
人工知能 11

ス

スタック 45
スタックポインタ 49
ストローブ信号 54

セ

正規化 43
セーブ(PCGの) 205
セーブ(PSGの) 208
1500-BASIC 70, 72
セントロニクスインターフェイス 183

ソ

双方向性データバス 48
ソフトウェア 26, 34
ソフトウェアスキャンニング 181

チ

チップセレクト管理 65
チャネルA 171
チャネルB 171
中央処理装置 11, 29
中間コード 41, 70

ツ

通信インターフェイス 15

テ

デイジーチェイン 52
デイジーチェイン割込み 59
ディスプレイ 22, 31
データ 34
データ書込み 157
データクロック 171
データ転送速度 57
データ転送プログラム 187
データ入力／出力 62
データバス 47
データファイル 174
データブロック 166
データマーク 174
テキストエリア 154
デモエース 207
テレビ表示方式 160

ト

同期通信方式 60
トーン発生器 65, 188
ドットマトリクスプリンタ 32
ドットモード 202
トランジスタ 9

ニ

2進法 29, 42
入出力装置 12
入出力端子 21
入出力ポート 51
入力装置 28, 30

ネ

音色 194

ノ

ノイズの設定 189
ノイズ発生器 65, 188
ノイマン型 9, 34

ハ

ハードウェア 152
ハードディスク 32
ハイインピーダンス 158
バイシンク 173
バイト 29
バイナリ同期通信モード 57
パソコンのアーキテクチャ 28
パターンエース 200, 201
パターンエースのコマンド 205
8253 62, 152, 192
8253の最大定格 63
8253の動作 63
8253の特徴 63

8253のピン配置 63
8255 61
8255周辺回路 182
8255の定格 61
8255の入出力ポート 61
8255のピン配置 61
8255のプログラミング 62
8ビットCPU 46
パレット回路 164
バンク管理 65
ハンドシェイク線 50, 51
汎用レジスタ 47

ヒ

ピアノ音のエンベロープ 209
ピークシフト 173
ビット 29
ビット制御 51
ビット同期 172
非同期通信方式 59
非同期通信モード 57
表示制御 154

フ

ファイルディスクリプタ 70
ファイルのSAVE 166, 179
ファイルのLOAD 179
ファイルブロック 173
フェッチサイクル 48
フォーマット(クイックディスクの) 166
フォント 202
フォントサンプル 211
複数ファイル管理方式 165
符号ビット 43
負数 43
浮動小数点内部表現 41
プライオリティ回路 162
プライオリティ設定回路 162
プラグボード方式 9
フリーエリア 70
プリンタ 25, 31, 152
プリンタインターフェイス 182
プリンタスプール 23, 33, 36
プリンタのリセット 188
プリンタバッファ 33, 186
プリンタルーチン 186
プレイミュージック 208
フレーム 202
プログラマブルカウンタ/タイマ 62
プログラミング言語 36, 37
プログラム内蔵方式 8, 34
ブロックファイル 166
フロッピーディスク 32
プロトコル 15
分周比 193

ヘ

ベリファイ 167
変数テーブル 44
変数名 44

ホ

ボイスカード 153
ボイス部 153
ボイスボード 25
ボードコンピュータ 15
ポート入出力 51
保護回路(クイックディスクの) 168
補助記憶装置 32
ボンディングオプション 57

マ

マイクロコンピュータ 11, 46
マイクロプロセッサ 13
マイクロプロセッサの開発過程 14
マスク 55
マスクデータ 200
マップ 201
マップフル 205
マップモード 203
マルチタスク 36

ミ

ミュージックエース 206
ミュージック演奏 65

メ

メモリ 29
メモリ空間 48
メモリコントローラ 65
メモリマップ 38, 67
メロディのくり返し 208

モ

モータコントロール 180
モード制御用レジスタ 51
モニタ 35, 38, 139, 154, 174
モニタコーリングシーケンス 144
モニタコマンド 39, 40
モニタサブルーチン 139, 144
モニタのワークエリア 140

ユ

ユーザープログラムエリアの拡張 206
優先順位 55

ヨ

読出しシーケンス(RAMファイルの) 155
4色カラープロッタプリンタ 25

ラ

ランダムアクセスメモリ 30

リ

リードオンリーメモリ 30
リセット 52
リセットスイッチ 42
リトライ(クイックディスクの) 167
リフレッシュ 65, 154, 157
リフレッシュサイクル 49

レ

レイトライトサイクル 158
レコーダ 22
レジスタ 47

ロ

6重和音 206

ワ

割込みアクノリッジ 55
割込みサイクル 55
割込み処理 192
割込み制御語 54
割込みベクトル 52, 53
ワンボードマイコン 16

A

ASCIIコード 30
AUTO RUN 44

B

BASIC 70
BASICインタープリタ 39
BASICコマンド 70
BASIC中間コード表 73
BCDカウント 64

C

Central Processing Unit 11, 29
CG ROM 154
CMT 153
CPU 11, 29
CRTコントローラ 65, 160
CRTC 65, 160
CRTCの基本ブロック 65

D

DIOのパラメータ 175

E

ENIAC 8, 34
Ent reg 139
Ext reg 139

F

FM記録方式 169
Frequency Modulation 169
Function 139

G

GDC 65

H

HDLC/IBM-SDLCモード 57

I

I/Oアドレスデコーダ 155
IC 11
IO空間 48
IORQ信号 53
IPL 37, 74

J

JOY部 154

K

KEY部 153

L

LIFO (last-in-fast-out) 49
LOAD (ファイルの) 167
LPT部 152
LSI 13, 46

M

M1信号 53
MCS-4 11
MFM記録方式 169
Microprocessor 29
Modified Frequency Modulation 169
MONITOR ROM 154
Moor's curve (曲線) 46
MPU 29
MZ-1500の特徴 19
MZ-1500のブロック図 152
MZ-1500のメモリ構成 20
MZ-40K 16
MZ-700 18, 70
MZ-80B 18
MZ-80C 16
MZ-80K 15, 16
MZ仕様とセントロニクス仕様(プリンタ) 184

N

NTSC方式 160

O

OS (Operating System) 35

P

PCG 65
PCG EDIT 201
PCG RAM 154
PCGエディタ 200, 201, 206
PCG画面 154
PCGコントローラ 154
PCGの重ね合せ 196
PCGのセーブ 205
PCGフォントミックス 197
PIO (Parallel Input/Output Interface) 49
PIO (Z80 PIO) 192
PIOのアーキテクチャ 50
PIOのアプリケーション 186
PIOの定格 50
PIOの動作モード 49, 53
PIOのピン配置図 50
PIOのプログラミング 52
Programable Sound Generator 21, 188
PSG 21, 65, 152, 188
PSG EDIT 206
PSGの構成 188
PWM (Pulse Width Modulation)方式 180

Q

QD 18, 32, 152, 165
QDインターフェイス回路 169
QDの概略仕様 165
QDの初期化 201
QDの内部回路 167
QDのファイルフォーマット 173
QDハンドラー 174, 176

R

RAM 30, 154
RAMファイル 22, 23, 32, 154, 155, 156
RAMファイルの書込みシーケンス 155
RAMファイルの読出しシーケンス 155
ROM 30
RS-232Cインターフェイス 22
Rsv reg 139

S

S-BASIC 70, 72
SDLC (Synchronous Data Link Control) 通信方式 60
Serial Input/Output Controller 56
SGC 188
SIO (Z80 SIO) 56
SIOの定格 58
SIOの特徴 57
SIOのピン配置 57
SIOの割込み 57
SN76489 65
SN76489の特徴 66
SN76489AN 188
SN76489ANのピン配置 66
SOUND部 152
Sound Generation Controller 188
SYSTEM-RAM部 154

T

TIMER部 152

V

VDP 65
VRAM 154
Von Neumann 9

Z

Z80CPU 47
Z80CPUの機能 49
Z80CPUの特徴 47
Z80CPUのピン配置図 48
Z80 PIO 183
Z80 SIO 56

シャープMZ-1500テクニカルノート
（パソコンの仕組みを探る）

定価2,500円

昭和59年12月15日 初 版 発 行

検印
省略

©監修 日本マイコンクラブ
編集人 鈴 木 勇 治
発行人 金 井 稔

発行 ラジオ技術社
〒101 東京都千代田区神田淡路町2－1
（電話03-251-0498，振替・東京5-2506）

ラジオ技術選書

101	オーディオ・テクニカル・マニュアル —コンポーネント・ステレオ・ガイド—	トリオ 編	定価1600円
102	エレキテル100章 —平賀源内からアポロまで—	柳沢 弦 著	定価 600円
103A	最新・オーディオ事典〈増補改訂版〉	パイオニア編集委員会 編	定価2800円
104	最新・カラーテレビ修理事典	平沢 進 著	定価2600円
105	ディジタルIC実用回路マニュアル	横井与次郎 著	定価2800円
106	リニアIC実用回路マニュアル	横井与次郎 著	定価3200円
107	波形観測〈オシロスコープ・テクニック・ガイド〉	ソニー・テクトロニクス 編	定価2600円
108	長岡鉄男／図解スピーカ	長岡鉄男 著	定価1600円
109	CMOSディジタル回路マニュアル	横井与次郎 著	定価2600円
110	最新・電子デバイス事典	馬場玄式 著	定価4300円
111	エレクトロ＝メカニカル機器	橋本順次 著	定価3500円
112	キットでおぼえる自作オーディオ	坂道 泰 著	定価1400円
113	マイクロコンピュータ基礎技術マニュアル	横井与次郎 著	定価3200円
114,115	スピーカ・システム（上），（下）	山本武夫 編著	（上）定価2600円 （下）定価2800円
116	マイコン技術教科書	橋本順次 著	定価3300円
117	FMチューナ・マニュアル	黒川 晃 著	定価3300円
118,119	マイクロコンピュータ活用マニュアル（上），（下）	横井与次郎 著	（上）定価3200円 （下）定価3400円
120	レコードとレコード・プレーヤ	井上敏也 監修	定価3500円
121	新しいオーディオ・アンプ	木塚 茂 著	定価2600円
122	マイクロコンピュータ・ソフトウエア基礎技術	横井与次郎 著	定価3800円
123	マイコン操縦法入門〈PC-8001を使って〉	平沢 進 著	定価2200円
124	プロッタ入門 —マイコンで図形を描く—	中村一治 監修	定価2600円
125	オーディオクラフト事典	入江順一郎 著	定価2600円
126	盤 塵 集 —音の姿を求めて—	池田 圭 著	定価2500円
127	マイクロコンピュータ・ハードウエア基礎技術	横井与次郎 著	定価3600円
128,129	詳解・ディジタルIC回路（上），（下）	後藤公雄 著	（上），（下）各 定価2300円
130	マイコン／ビジネス・ソフト1年生	西尾文明 著	定価2200円
131	ディジタル・オーディオ —基礎理論と最新技術—	土井利忠・伊賀 章 著	定価3300円
132	アップル操縦法入門 —21世紀への知的自転車—	近藤龍太郎 著	定価2600円
133	FM8操縦法入門 —パソコンとのつきあい方—	脇 英世 著	定価2400円
134	パソコン・グラフィックスの基礎 —主要機種で画を描くテクニック集—	矢矧晴一郎 著	定価2600円
135	パソコン操典 —戦う企業の講習テキスト—	住友ゴム工業KK 編著	定価2000円
136	プロならこうする，パソコン・プログラミング —MZシリーズを使って—	宮永好道 編著	定価2400円
137	BASICと比較したPASCAL入門	横井与次郎 著	定価4300円
138	プリンタ操縦法入門	脇 英世 編著	定価2400円
139	パソコン／メカトロニクス実験室	岡田・高橋・永井・松田 共著	定価2400円
140	ハンドヘルド・コンピュータHC-20大事典	平林千春・須藤慎一 著	定価2900円
141	マイクロコンピュータ制御基礎技術	佐々木 正 監修	定価3400円
142	BASICによる作表とモジュール化高速グラフィックス	横井与次郎 著	定価2200円
143	PC-9801シリーズ操縦法入門（グラフィックス/ディジタル処理/日本語ワープロ）	三谷政昭 著	定価2400円
144	親子で楽しむMSX	上柿 力 著	定価1800円
145	PC-98/88 BASIC高速グラフィックス —基礎編—	横井与次郎 著	定価2600円
146	おもしろ・パソコン・テクニック おたのしみプログラムの作り方	須藤慎一 著	定価2600円
147	HC-88操縦法入門 —情報化時代のサバイバル・ツール—	片岡寛秀 著	定価2400円
148	FM-77の本 —BASIC, Logoプログラミング—	上柿 力 著	定価2200円
149	PC-9800シリーズ ビギナーズガイド —システムの組み方から活用まで—	マイクロデザイン 編著	定価1800円
150	Logo 人工知能へのアプローチ	祐安重夫 著	定価2400円
151	MZ-1500テクニカルノート —パソコンの仕組みを探る—	日本マイコンクラブ 監修	定価2500円

mz-1500 テクニカルノート

ラジオ技術社　　定価2,500円

ISBN4-8443-0151-9 C2055 ¥2500